View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Savoirs UdeS

SUR L'UTILISATION DE L'APPRENTISSAGE PROFOND POUR LA PRÉVISION PROBABILISTE EN SÉRIES CHRONOLOGIQUES

par

Adam Salvail

Mémoire présenté au Département d'informatique
en vue de l'obtention du grade de maître ès sciences (M.Sc.)

FACULTÉ DES SCIENCES

UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, 19 janvier 2021

Le 19 janvier 2021

Le jury a accepté le mémoire de Adam Salvail dans sa version finale

Membres du jury

Professeur Hugo Larochelle
Directeur
Département d'informatique

Professeur Jean-Pierre Dussault
Codirecteur
Département d'informatique

Professeur Nocolas Chapados
Codirecteur
Département d'informatique

Professeur Pierre-Marc Jodoin
Membre interne
Département d'informatique

Professeur Marc Frappier
Président-rapporteur
Département d'informatique

Sommaire

L'utilisation de l'apprentissage profond dans le domaine des séries chronologiques commence tout juste à faire son apparition. Bien qu'historiquement cette technique était considérée comme trop complexe, la tendance courante des mégadonnées permet maintenant d'appliquer de façon appropriée des modèles comme les réseaux de neurones profonds au problème de la prévision de séries chronologiques. Fort de cette réalisation, ce texte explore l'utilisation de l'apprentissage profond dans l'objectif de caractériser l'incertitude des prévisions produites par le modèle, une tâche particulièrement adéquate pour ces modèles à capacité élevée.

Mots-clés: apprentissage profond ; séries chronologiques, prévisions probabilistes, réseaux de neurones récurrents, régression quantile

Remerciements

La réalisation de ce mémoire a été possible grâce au support de plusieurs personnes à qui j'aimerais dédier ces remerciements.

Je tiens d'abord à adresser toute ma reconnaissance à mes trois codirecteurs de recherche, Hugo Larochelle, Jean-Pierre Dussault et Nicolas Chapados pour m'avoir fait découvrir les domaines de l'apprentissage automatique, de l'optimisation mathématique et de la prévision en séries chronologiques, respectivement. Ils ont tous les trois pu me servir de guide dans ces domaines dynamiques de la recherche et ont chacun facilité mon initiation à ces domaines académiques.

Je désire aussi remercier mes collègues du SMART, du BISOUS et d'Element AI pour toutes les discussions passionnantes et les diverses collaborations qui m'ont aidé à pousser mes réflexions au prochain niveau.

J'adresse également mes sincères remerciements à tous les professeurs, techniciens et professionnels des départements d'informatique et de mathématiques de l'Université de Sherbrooke qui à travers leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé ma formation universitaire et ont contribué à la personne que je suis aujourd'hui.

Je veux également remercier chaleureusement Ignatius qui m'a apporté son soutien moral et intellectuel tout au long de la rédaction.

Je souhaite finalement témoigner toute ma gratitude envers ma mère en remerciement pour tous les sacrifices qu'elle a faits pour me donner la chance d'en arriver à cette étape.

Abréviations

ARIMA Auto-Regressive Integrated Moving Average process

ARMA Auto-Regressive Moving Average process

DeepAR Deep Auto-Regressive network

EAM Erreur Absolue Moyenne

EAMC Erreur Absolue Moyenne Calibrée

EAMP Erreur Absolue Moyenne en Pourcentage

ELU Exponential Linear Unit

EQM Erreur Quadratique Moyenne

FFNN Feed-Forward Neural Network / Réseau de neurones profond simple

LSTM Long Short Term Memory Network

M4 Quatrième compétition organisé par Makridakis

MQ-RNN Multi-horizon Quantile Recurrent Neural Network

PReLU Parametric Rectified Linear Unit

ReLU Rectified Linear Unit

ResNet Residual Network

Seq2Seq Sequence to sequence / réseau encodeur-décodeur

Table des matières

Liste des figures

Liste des tableaux

Introduction

À l'été 2017, l'*International Symposium on Forecasting* avait lieu ; une conférence sur le sujet de la prévision de séries chronologiques. Celle-ci se déroulait en juillet à Boulder, au Colorado, ce qui explique surement pourquoi il faisait aussi chaud et humide.

Habitué aux conférences en apprentissage automatique comme NeurIPS et ICLR, l'ISF était comme une bouffée d'air frais : tout était différent ! Un fait impressionnant pour un habitué des conférences en apprentissage automatique était le nombre de participants. En tout et partout, la conférence recevait quelques centaines de personnes, une différence de taille quand on la compare aux milliers de personnes présentes à Neurips. Les entreprises commanditaires ne prenaient pas la moitié de l'espace, les repas n'étaient pas fournis... De loin le plus marquant était l'esprit de famille. Les gens présents à la conférence agissaient tous comme s'ils venaient visiter des cousins lointains, telles des retrouvailles organisées annuellement.

« N'avez-vous pas peur que l'IA prenne une grande part de marché dans votre domaine ? » fut la question innocemment demandée. L'interpelé éclata simplement de rire. Ceci n'est qu'un exemple d'un moment où il était évident que la communauté d'experts en séries temporelles nie tout simplement que l'apprentissage automatique puisse les aider d'une quelconque façon que ce soit.

L'organisateur des compétitions M, une série de compétitions internationales en prévision de séries chronologiques écrivait ceci en introduction à la quatrième itération de la compétition :

> « Machine Learning (ML) methods have been proposed in the academic literature as alternatives to statistical ones for time series forecasting. Yet, scant evidence is available about their relative performance in terms

> of accuracy and computational requirements. [...] After comparing the post-sample accuracy of popular ML methods with that of eight traditional statistical ones, we found that the former are dominated across both accuracy measures used and for all forecasting horizons examined. » [MSA18]

En fait, il se trouve que les compétitions M avaient toujours eu pour but de démontrer empiriquement que les modèles simples étaient en tout temps préférables aux modèles plus complexes en l'absence de plus d'information sur le problème [MH00], une hypothèse plutôt raisonnable. Or, ces compétitions ont eu assez d'influence pour que la majorité des articles publiés dans le domaine de la prévision de séries temporelles doive inclure des tests sur ces jeux de données. Le problème avec cet état de fait est que les séries qui s'y retrouvent ont historiquement été très limitées en diversité : elles étaient toutes des séries plutôt courtes (la plus longue série de la compétition M3 avait à peine plus de cent mesures), sans covariables, univariées et à une résolution très basse (la plus élevée étant des séries à valeurs mensuelles). Autrement dit, les données étaient particulièrement adaptées à utiliser des méthodes simples.

Plusieurs ont tenté d'entrainer des algorithmes tels les réseaux de neurones, sans succès [MSA18]. En plus du temps de calcul élevé (entrainer 3000 réseaux de neurones ne se fait pas en un clin d'œil), les performances n'arrivaient jamais à battre les modèles plus simples. Ceci n'est pas tellement étonnant lorsqu'on considère que souvent les architectures utilisées avaient plus de paramètres qu'il n'y avait de données utilisées pour entrainer les modèles. Bien qu'il ait été démontré que la surparamétrisation des modèles amène parfois une généralisation accrue [ACH18], ces travaux n'ont toujours pas été appliqués aux modèles récurrents.

Lors de la quatrième édition de la compétition, le vainqueur a utilisé une méthode plus tard qualifiée « d'hybride » combinant un LSTM (Section 2.4.1) à un algorithme de lissage exponentiel. Cette itération de la compétition était dotée d'un jeu de données de cent-mille séries chronologiques, une première. Or, la méthode gagnante a fait une percée intéressante : le LSTM est entrainé sur l'ensemble des séries d'une même résolution, pas pour chacune d'entre-elles. Ce faisant, le modèle d'apprentissage automatique avait enfin accès à assez de données pour pouvoir être entrainé profitablement.

D'ailleurs, il est à noter que la distinction entre les modèles dits « statistiques » et les modèles « d'apprentissage automatique » est purement arbitraire et prétend être basée sur la communauté d'où provient le modèle. Il pourrait être défendu que la distinction se rattache à la philosophie derrière le type de modélisation, mais mathématiquement, il n'existe absolument aucune différence entre les deux familles. Dans ce texte, le terme *apprentissage profond* est utilisé pour distinguer la capacité d'apprentissage plus élevée de ce type de modèle.

En 2019, un an après la fin de la quatrième compétition, une équipe d'Element AI a publié N-BEATS [OCCB20], un modèle pour lequel tous s'entendent pour dire qu'il repose entièrement sur l'apprentissage automatique. Ce dernier a battu la méthode hybride vainqueur de la compétition de façon considérable et a ramené l'apprentissage automatique sur les lèvres de la communauté de la prévision de séries chronologiques.

Le domaine des prévisions de séries chronologiques en entreprise a historiquement porté sur les prévisions d'assez faible fréquence. À ses balbutiements dans les années 50 [AHM51], puisque les ordinateurs n'étaient nullement aussi présents qu'aujourd'hui, la quantité de données qu'une personne pouvait analyser était assez limitée. Les économistes suivaient (et suivent toujours, dans certains cas) l'état de l'économie avec des mesures annuelles et trimestrielles ou, dans de rares cas, mensuelles. Par conséquent, la méthode de Holt-Winters [Bro04] dans les années 60 ou de Box-Jenkins [BJ70] dans les années 70 étaient des méthodes très appropriées et reste aujourd'hui deux modèles simples qui doivent servir de référence.

Cependant, avec l'évolution de la technologie et l'ère des mégadonnées, il est désormais normal de collecter des ensembles de données avec un niveau de granularité bien plus élevé, parfois même en temps réel. Par ailleurs, puisque la capacité de stockage des données est plus élevée, plusieurs sources de données connexes au problème deviennent accessibles. Ceci représente un changement de paradigme où les modèles plus simples n'ont simplement pas la capacité nécessaire pour faire des prévisions de qualité.

Par exemple, pour la prédiction de la demande d'un produit sur les trois prochains mois, il aurait historiquement fallu avoir accès au nombre d'articles vendus mensuellement, soit en agrégé ou ventilé par magasin. De ces données, il aurait été possible de déceler les tendances de ventes, de retirer l'effet des saisons et d'estimer la

variance des ventes pour ensuite en faire des projections sur les trois prochains mois. Loin d'être primitive, cette méthode donne en fait des résultats robustes, explicables et, si les données ne sont pas trop bruitées, exacts. Par contre, le prix à payer est en analyste-heures.

Or, pour être compétitif aujourd'hui dans le monde du commerce de détail, il faut optimiser ses processus de façon à couper le plus d'inefficacités possible. Dans un monde du « juste-à-temps », une prévision mensuelle n'apporte plus assez d'information.

De plus, la nature des questions change également. Plutôt que d'avoir une estimation du nombre de produits qui seront vendus, il peut être plus intéressant de savoir le jour où il ne restera plus de cet item dans un certain magasin. Dans ce cas, ce que l'analyste tente de prévoir n'est plus tant le stock disponible en magasin autant que la durée de temps avant une pénurie. Si un modèle de prévisions en continu est déjà en place, ces deux problèmes sont équivalents. Par contre, si tout ce qu'il existe est un modèle de prévision de la demande mensuelle, il faut complètement repenser sa stratégie et modéliser le temps entre certains évènements plutôt que le stock. L'approche de Croston [Cro72], une méthode similaire à ce qui vient d'être énoncé, a d'ailleurs été utilisée avec succès pour faire ce genre de prévisions.

Un problème ignoré jusqu'à maintenant est celui de l'exactitude des prévisions. Si un modèle avance que le stock sera épuisé le 14 mars, est-ce surprenant d'avoir une pénurie le 12 mars ? Le 21 mars ? Sans une mesure robuste de l'incertitude des prévisions, il devient difficile de prendre des décisions éclairées reposant sur les résultats. Même aujourd'hui avec les méthodes plus simples qui reposent déjà, pour la plupart, sur des méthodes probabilistes, il est difficile d'obtenir des intervalles de confiance à l'aide des logiciels courants de prévisions [1].

Finalement, un dernier problème des méthodes plus simples est tout simplement leur manque de perception globale [MMH20]. Pour revenir à l'exemple de prédiction de la demande d'un produit, si les données ventilées par magasin sont disponibles, il est bien plus simple d'appliquer une méthode simple à chacun des magasins, séparément. Y a-t-il de l'information qu'il serait possible d'extraire du fait que ceci est un même

1. Heureusement, ceci est en train de changer si on se fit au nombre grandissant de vendeurs qui louangent les prévisions probabilistes au *International Symposium on Forecasting*.

produit, offert à plusieurs endroits. Peut-être que l'un des magasins est un précurseur de la demande dans les autres, auquel cas cette information devient très importante dans la prise de décision. Malheureusement, partager cette information entre les séries n'est vraiment pas simple et requiert des modèles plus complexes.

Bref, les temps changent et l'époque où les prévisions étaient seulement utilisées comme un outil d'aide à la décision (ce qu'elles sont toujours aujourd'hui) a fait place à une époque où il devient nécessaire de produire des prévisions probabilistes utilisant d'énormes ensembles de données afin de minimiser les couts d'opération. Dans cette nouvelle réalité, il est nécessaire d'avoir recours à des modèles ayant un plus grand pouvoir de modélisation comme l'apprentissage profond.

Cette famille de modèles est la clé ouvrant la porte vers des méthodes qui supporte les grands ensembles de données avec plus de flexibilité. Ils sont à même de produire des prévisions probabilistes non paramétriques et, par leur grande capacité, peuvent être appliqués à l'ensemble des cas d'intérêt, ce qui évite le problème de manque de mémoire.

Ce document explore ces idées. D'abord, le Chapitre 1 décrit formellement ce qu'est une série chronologique et en quoi consiste la prévision. Ensuite le Chapitre 2 définit l'apprentissage automatique et tout particulièrement l'apprentissage profond et les réseaux de neurones. Le chapitre se termine en couvrant comment l'apprentissage profond peut être utile pour la tâche de prévision de séries chronologiques. Finalement, le Chapitre 3 décrit une sous-tâche de la prévision, la prévision probabiliste, ainsi que pourquoi l'apprentissage profond est particulièrement adapté à ce type de tâche. Le chapitre conclut avec une étude empirique de quelques méthodes pouvant être utilisées pour la prévision probabiliste en mettant l'accent sur l'utilisation de ces méthodes dans un contexte d'exploration des données, c'est-à-dire en évitant les modèles complexes efficaces que sur un type précis d'application.

Chapitre 1

Les séries temporelles

« Ceux qui ignorent l'histoire seront condamnés à la revivre. » Au coeur de cette maxime, on retrouve les éléments principaux qui ont mené à l'étude des séries temporelles, aussi appelées séries chronologiques.

D'abord, il y a l'histoire. Du point de vue des données, l'historique est constitué d'une suite d'instants au cours desquels on peut mesurer certains phénomènes d'intérêt. Un **phénomène** est n'importe quel système, réel ou virtuel, qui évolue de façon systématique à travers le temps et une **mesure** est une valeur numérique réelle ou entière qui décrit le phénomène. De la température à Sherbrooke à la population de la Chine, du prix de l'action d'Alcan au niveau de la mer Méditerranée, on souhaite préserver une trace d'un phénomène au cours du temps en enregistrant ces mesures. Une **série chronologique** est une séquence de telles mesures, souvent enregistrées dans le but de mieux comprendre le phénomène ou d'en dériver un modèle mathématique. Plus il faut retourner dans le passé pour comprendre le phénomène, plus l'historique des valeurs est important.

Ensuite, il y a le concept de « revivre l'histoire ». Le phénomène suivi à travers le temps doit être assez régulier afin de pouvoir le décrire avec un certain degré de précision sans devoir énumérer toutes les valeurs enregistrées. En d'autres mots, il est supposé qu'il soit possible de définir un modèle mathématique qui décrive le phénomène observé avec une erreur de modélisation (voir la définition Section 3.1) finie. Or, s'il est possible de caractériser la valeur de la série, il devient possible de détecter les mesures anormales et de prévoir, avec un certain niveau de certitude,

les mesures que prendra la série dans le futur. Ce document s'attarde à la seconde partie : la **prévision** de séries chronologiques.

Les mesures constituant une série chronologique ont certaines caractéristiques qui sont particulièrement importantes telles que leur résolution, leur domaine ou l'absence potentielle de certaines mesures au fil du temps. La **résolution** des mesures se caractérise par l'espace temporel les séparant. Si les mesures sont prises en continu, la série chronologique devient un **signal**. Ce texte ne s'intéresse qu'aux séries temporelles ayant des mesures prises à intervalles discrets et il est usuel de préciser le niveau de granularité de la série (ex. : horaire, mensuelle, annuelle...). À moins d'indication contraire, ce document suppose que ces intervalles sont réguliers, c'est-à-dire que le temps entre chaque mesure reste le même, et qu'il n'y a pas de mesures manquantes. Le **domaine** des mesures est normalement présumé être l'ensemble des réels, mais il n'est pas rare de rencontrer une restriction aux valeurs positives où à des valeurs entières.

Ce chapitre décrit en un peu plus de détails ce qu'est une série chronologique en abordant ses caractéristiques les plus fréquemment discutées.

1.1 Définitions et notations

Afin de correctement prendre en compte l'incertitude entourant la modélisation d'un phénomène, l les séries temporelles sont modélisées de façon probabiliste. Soit Y_t la variable aléatoire décrivant une mesure prise sur un phénomène d'intérêt au temps $t \in \mathbb{N}$, $\{Y_t\}_{t \in T \subseteq \mathbb{N}}$ dénote une série de variables aléatoires appelées **processus stochastique** qui est l'outil probabiliste[1] représentant une série temporelle. Afin d'alléger la notation, $Y_{a:b} = \{Y_t\}_{t \in \{a, a+1, \ldots, b\}}$ est utilisé pour représenter un intervalle de la série pour les **pas de temps** entre a et b, inclusivement.

Une série chronologique est dite **autorégressive** lorsque Y_t peut être expliqué (du moins en partie) par les valeurs $Y_{1:t-1}$ appelées l'**historique** de sorte qu'il existe

1. Formellement, soir un espace probabilisé $(\Omega, \mathcal{F}, P)$, respectivement l'univers d'échantillonnage Ω, la σ-algèbre $\mathcal{F}$ sur Ω et la mesure P définit sur $\mathcal{F}$ tel que $P(\varnothing) = 0$ et $P(\omega) = 1$. À ceci est ajouté un ensemble d'indices $T \subseteq \mathbb{N}$ muni d'un ordre total et d'un espace d'état $S \subseteq \mathbb{R}$. Un processus stochastique est une collection de variables aléatoires définies sur un même espace de probabilité $(\Omega, \mathcal{F}, P)$, indexée par T à valeurs dans S.

une fonction f telle que $Y_t = f(Y_{1:t-1}, \varepsilon_t)$ où ε_t est appelé l'**innovation** et représente l'information qui ne peut être capturée par f. Par exemple, avec un modèle linéaire à coefficients $a_i \in \mathbb{R}$ et une innovation additive, il se trouve que $Y_t = \sum_{i=1}^{p} a_i Y_{t-i} + \varepsilon_t$. Le caractère autorégressif d'une série temporelle est utile pour effectuer des **prévisions**, c'est-à-dire de modéliser les valeurs du passé afin d'extrapoler les valeurs de la série au pas de temps $Y_{t+1:\infty}$ (le **futur**). Habituellement, ces prévisions sont requises sur un certain **horizon** de temps h. Le problème devient alors d'estimer $p(Y_{t+1:t+h}|Y_{1:t})$.

Un phénomène, en plus de dépendre de ses états passés, peut également être influencé par d'autres stimulus. Dans le langage des séries chronologiques, les mesures de ces autres stimulus sont appelées **covariables**[2]. Les **covariables intemporelles** sont les valeurs $x_0 \in \mathbb{R}^n$ qui n'évoluent pas dans le temps et qui ont une influence sur la série chronologique dans son ensemble. Les **covariables temporelles** sont quant à elles les valeurs $x_{1:t} \subset \mathbb{R}^m$ qui évoluent dans le temps et influencent les valeurs de $Y_{1:t}$. De plus, ces dernières doivent également être séparées entre les covariables dont la valeur est connue dans au futur $t + h$ et celles où la valeur reste inconnue. Si les valeurs de $x_{t:t+h}$ sont déjà connues, elles peuvent donc être utilisées pour la modélisation des prévisions.

Une série chronologique est dite **univariée** si l'image de Y_t est un espace unidimensionnel ou **multivariée** quand Y_t a pour image un espace vectoriel. Pour le reste de ce document, il est supposé que la série est limitée au cas univarié puisque la définition de quantiles, abordée au Chapitre 3, ne peut pas être facilement étendue à toutes les distributions dans le cas multivarié.

Une propriété très importante des séries chronologiques porte sur la stabilité temporelle de la distribution de probabilité du processus stochastique $Y_{1:t}$. Si la distribution de probabilité de laquelle Y_t est échantillonnée reste la même au fil du temps, le processus stochastique lié à la série chronologique est dit **stationnaire** [BDC02]. Cette propriété est ce qui permet de faire de l'extrapolation. En l'absence de valeurs explicatives, si la série n'est pas (ou ne peux pas se réduire à) une série stationnaire, il devient très difficile de produire des prévisions à partir des valeurs observées.

2. Ce texte suppose que les mesures des phénomènes sont exactes et ne sont donc pas des variables aléatoires, dénotées par des majuscules. Le travail pourrait être généralisé de sorte que même les variables en entrée soient considérées aléatoires.

Bien entendu, les séries chronologiques sont rarement naturellement stationnaires. Le problème le plus fréquent est la présence d'une **tendance** (qui peut être positive ou négative) qui poussent les valeurs de la série à la hausse ou à la baisse, ce qui fait en sorte que la moyenne de la série se trouve à changer à travers le temps. Ceci est généralement modélisé par une fonction linéaire, quadratique ou exponentielle, mais en réalité, n'importe quelle fonction continue, monotone ou non, peut faire l'affaire. La **saisonnalité** est une variation spécifique de la tendance caractérisée par son effet oscillatoire régulier sur les valeurs de la série, comme l'effet des saisons sur la température. Si l'effet oscille, mais n'est pas régulier, il est question d'un **cycle**.

Une version relaxée de la stationnarité est le concept de la **scédasticité** qui caractérise la variance de l'incertitude aléatoire, c'est-à-dire la variance inhérente à la série [3]. Une série chronologique est **homoscédastique** si la variance est constante à travers le temps et **hétéroscédastique** dans le cas opposé.

1.2 Prévisions

Extrapoler des mesures décrivant un phénomène vers le futur est un exercice difficile, mais ayant beaucoup de valeur pour informer la prise de décisions. Bien que l'omniscience reste hors de portée, être plus informé et bien utiliser les informations à sa disposition pour venir en aide à la prise de décision peut faire la différence entre le succès et l'échec.

Pour y arriver, un analyste construit un **modèle** de la série chronologique qui tente de résumer de façon mathématique les principes sous-jacents menant aux mesures enregistrées. Or, les séries chronologiques rencontrées diffèrent selon leur provenance. En ingénierie, les séries sont souvent à indices continus, ce qui requiert un tout autre type de modèle que la finance ou le commerce où les séries sont à indices discrets. Même dans ces derniers cas, la différence de résolution entre la finance où il y a une abondance de mesures et le commerce où les stocks ont tendance à être analysés à une fréquence beaucoup plus basse oblige l'utilisation de différents types de modèles. Plus la résolution de la série est basse, moins il y a de données sur lesquelles baser les

3. Voir la Section 3.1 pour une définition plus détaillée.

prévisions, moins le modèle peut être complexe, sous peine d'être surentrainé [4]. De plus, plus la période considérée est longue, moins y il y a de chances que les facteurs causant le phénomène restent stables, un problème appelé la **dérive conceptuelle**.

Un modèle f de série chronologique tente d'approximer les valeurs futures de la série $y_{1:t}$. En d'autres termes, le modèle produit des prévisions $\hat{y}$ de sorte que

$$y_{t+1} = \hat{y}_{t+1} + \varepsilon = f(y_{1:t}) + \varepsilon_t$$

où ε est l'erreur de modélisation commise par le modèle, parfois appelé l'**innovation** [5].

Pour y arriver, un analyste choisit une famille $\mathcal{M}$ de modèles paramétrés par θ de sorte que

$$\mathcal{M} = \{f(\cdot;\theta) \mid \theta \in \Theta \subset \mathbb{R}^d, d \in \mathbb{N}\}.$$

Avec le bon choix de θ, il est possible d'utiliser le modèle pour produire des prévisions. Les techniques d'approche de θ sont discutées en plus de détails dans la Section 2.3.

1.2.1 Prévisions directes et prévisions récursives

Une particularité de la modélisation de séries chronologiques par rapport aux modèles typiquement créés en apprentissage automatique est l'utilisation répétée du modèle pour une même série. En effet, il est plutôt rare de vouloir des prévisions que pour le prochain pas de temps. En général, un analyste souhaite créer des prévisions couvrant un horizon de prévisions. Pour y arriver, il y a deux options.

Les **prévisions directes** utilisent le modèle pour faire de multiples prévisions à la fois, c'est-à-dire que le modèle génère plusieurs valeurs, chacune pour un pas de temps différent au fil de l'horizon. Mathématiquement, f est changée pour

$$y_{t+1:t+h} = \hat{y}_{t+1:t+h} + \varepsilon_{t+1:t+h} = f(y_{1:t};\theta) + \varepsilon_{t+1:t+h}$$

où ε_t peut avoir une valeur constante ou dépendante du pas de temps de la prévision. Cette façon de faire est la plus simple, surtout en conjonction avec les modèles

4. Voir Section 2.2 pour une définition du surentrainement.

5. À noter que certains préfèrent, contrairement à cet auteur, définir l'innovation comme étant $\varepsilon_t = \hat{y}_{t+1} - y_{t+1}$. Voir [GTS16] pour une discussion des pour et contre de chacune des notations.

d'apprentissage automatique usuels et fonctionne particulièrement bien si le modèle est utilisé une seule fois au pas de temps t et inclue toutes les covariables connues, passées ou futures.

La seconde option est d'utiliser les **prévisions récursives** qui sont créées à partir d'un modèle ne prédisant que la prochaine valeur y_{t+1} de la série et de l'appliquer à répétition, en utilisant les valeurs prédites comme nouvelles entrées, jusqu'à ce que le modèle ait produit les h prévisions voulues.

$$\begin{aligned} y_{t+1:t+h} &= \hat{y}_{t+1:t+h} + \varepsilon_{t+1:t+h} \\ &= [f(y_{1:t}, \varnothing; \theta), f(y_{1:t}, \hat{y}_{t+1}; \theta), \ldots, f(y_{1:t}, \hat{y}_{t+1:t+h-1}; \theta)] + \varepsilon_{t+1:t+h} \end{aligned}$$

Cette façon, plus intuitive, permet de reproduire l'utilisation potentielle du modèle où à chaque pas de temps, une nouvelle prévision est créée. Le plus grand défi de cette méthode au moment de l'entrainement est de s'assurer de bien départir les valeurs connues des valeurs inconnues au temps de la prévision afin de ne pas polluer les données d'entrainement avec de l'information qui serait inconnue lors de l'application du modèle. Il faut d'ailleurs apporter une attention toute particulière pour les covariables liées aux temps futurs.

La décision d'utiliser des prévisions directes ou récursives est sans doute la plus importante au niveau du choix du modèle et dépend directement de la façon dont les prévisions seront consommées. Pour les indécis, il existe certains modèles comme les réseaux de neurones récurrents, abordés à la Section 2.4, qui sont en mesure de combiner les deux méthodes, c'est-à-dire de faire des prévisions directes utilisées de façon récursive pour être mises à jour. Une étude plus approfondie de la question est réalisée par [TH14].

1.2.2 Mesures de performance

Une fois qu'un modèle a créé ses prévisions, il est de bon ton de vouloir évaluer ses performances afin de s'assurer de pouvoir faire confiance aux prévisions. Cette section décrit les mesures de performance les plus couramment utilisées dans la littérature.

Soit $\{y^{(i)}_{1:t+h}\}_{i \in 1:N}$ un ensemble de N séries chronologiques. La tâche est de créer des prévisions $\hat{y}_{t+1:t+h}$ pour les h pas de temps suivants t pour chacune des N séries

chronologiques constituant l'ensemble de données à l'étude.

D'abord, il est toujours possible d'utiliser l'**erreur absolue moyenne** ou l'**erreur quadratique moyenne** :

$$EAM(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h} \sum_{j=t+1}^{t+h} |y_j^{(i)} - \hat{y}_j^{(i)}|$$
$$EQM(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h} \sum_{j=t+1}^{t+h} (y_j^{(i)} - \hat{y}_j^{(i)})^2.$$

Par expérience, l'EQM a tendance à être utilisée pour l'entrainement comme fonction de cout alors que l'EAM est utilisée comme mesure de performance due à sa plus simple interprétation de la déviation moyenne [6].

Dans tous les cas, ces deux mesures ont un problème majeur : elles mettent l'accent sur les séries chronologiques de l'ensemble de données ayant une plus grande magnitude, ce qui est un problème autant en entrainement que pour l'évaluation du modèle. Pour pallier ce problème à l'évaluation, il suffit de normaliser la mesure, d'où l'**erreur absolue moyenne en pourcentage** :

$$EAMP(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h} \sum_{j=t+1}^{t+h} \left| \frac{|y_j^{(i)} - \hat{y}_j^{(i)}|}{y_j^{(i)}} \right|.$$

Cette mesure permet de plus simplement comparer la performance d'un modèle sur des séries temporelles de magnitudes différentes, mais a deux problèmes : il faut qu'aucune valeur ne soit égale à zéro et, dû à la façon dont les pourcentages sont calculés, cette mesure a tendance à plus pénaliser numériquement les prévisions qui sous-estiment la valeur de la série, ce qui est commun des mesures de performances relatives.

Une autre méthode populaire est l'**erreur absolue moyenne calibrée** qui normalise l'erreur de prévision par l'erreur de prévision qui aurait été commise par un

6. Afin de rendre l'EQM plus facile d'interprétation, il n'est pas rare de rencontrer sa mesure cousine : la racine carrée de l'EQM.

modèle naïf [7] :

$$EAMC(Y, \hat{Y}) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{h}\sum_{j=t+1}^{t+h}\frac{|y_j^{(i)} - \hat{y}_j^{(i)}|}{\frac{1}{t-1}\sum_{k=2}^{t}|y_k^{(i)} - y_{k-1}^{(i)}|}.$$

Dans le cas d'une série où il y a une saisonnalité apparente de longueur $s < t$, le modèle naïf est ajusté pour utiliser la valeur de la période précédente, c'est-à-dire

$$EAMC(Y, \hat{Y}) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{h}\sum_{j=t+1}^{t+h}\frac{|y_j^{(i)} - \hat{y}_j^{(i)}|}{\frac{1}{t-1}\sum_{k=s+1}^{t}|y_k^{(i)} - y_{k-s}^{(i)}|}.$$

En cas de saisonnalité forte, cette dernière version offre généralement une comparaison plus juste puisque le modèle naïf saisonnier affiche de meilleures performances. De plus, cela permet d'avoir des prédictions naïves qui ne soient pas constantes, mais qui varient au long de la saison. Par contre, si les séries disponibles sont plutôt courtes, c'est-à-dire que la longueur de la saison prend une partie importante de l'ensemble des données disponibles voire qu'une saison complète ne soit pas disponible, il faudra se rabatre sur la version originale de la mesure.

Il n'y a pas vraiment de méthode généralement reconnue comme étant supérieure, malgré certains travaux ayant pour but d'évaluer la question [HK06]. Malgré tout, ce document fait le choix arbitraire de se concentrer sur l'EAMC comme mesure de performance puisqu'elle évalue naturellement la médiane des distributions des prévisions.

1.3 Les grands ensembles de données

Le premier cas où l'apprentissage profond peut vraiment aider les méthodes de prévision est en présence d'une grande quantité de données. Les méthodes statistiques couramment utilisées en prévision de séries temporelles reposent sur des modèles linéaires probabilistes, ce qui les rendent à la fois trop limitées en termes de puissance de représentation, appelée **capacité**, ou trop complexes en termes de calcul quand ils

7. Le modèle naïf propose la dernière valeur connue de la série temporelle comme prévision. Par conséquent, son erreur en cas de prévisions récurrentes est définie par la différence entre les valeurs de la série : $\frac{1}{t-1}\sum_{k=2}^{t}|y_k^{(i)} - y_{k-1}^{(i)}|$.

sont plus élaborés (par exemple en nécessitant d'inverser des matrices ou d'utiliser de l'échantillonage Monte-Carlo). Cette complexité de calcul croît généralement selon le nombre de données ou le nombre de paramètres à estimer [8].

Une série chronologique a généralement trois axes de croissance en volume de données :

— la résolution de mesure, c'est-à-dire le temps qui s'écoule entre les valeurs de la série ;

— le nombre de séries mesurées en parallèle, par exemple, si un analyste a plusieurs produits pour lesquels il souhaite prédire la demande ;

— la quantité d'information exogène, c'est-à-dire les données explicatives qui ne sont pas l'objet de l'étude, mais qui permettent de faire de meilleures prévisions.

Chacun de ces axes de croissance des données apporte ses propres couts et bénéfices. L'apprentissage profond permet généralement d'exploiter la plus grande quantité de données avec ses modèles dotés d'une capacité plus élevée.

Un quatrième axe, non présenté ici, concerne la longueur de l'historique, c'est-à-dire la durée de temps où des valeurs historiques sont disponibles. Bien que lié au domaine d'application, il est généralement reconnu qu'il y a un gain décroissant à allonger l'historique des données disponibles. En fait, dans la plupart des cas, l'historique est coupé ou remplacé par des mesures de tendances. Pire, il est plutôt rare qu'un processus stochastique soit stationnaire sur une si longue période (les humains ont la fâcheuse tendance à s'adapter à leur environnement et à chercher à l'améliorer), rendant les mesures du passé lointain dangereuses pour la modélisation. L'utilisation d'un plus long historique afin d'augmenter sa confiance en un modèle doit être faite avec prudence, car cette stratégie risque plutôt de biaiser le modèle sur des données n'étant plus tout à fait représentatives de la situation future.

8. Ceci suppose que le nombre de données soit proportionnel à la complexité du signal présent dans les données, ce qui n'est pas nécessairement le cas.

Chapitre 2

L'apprentissage automatique

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle dont le but est de créer des modèles qui arrivent à apprendre d'eux-mêmes à résoudre un problème plutôt que de dépendre de la construction méticuleuse d'algorithmes par des experts. L'apprentissage automatique se décline en trois grandes familles : l'apprentissage par renforcement, l'apprentissage non supervisé et l'apprentissage supervisé.

Le plus important pour ce document est l'apprentissage supervisé. Ce dernier cherche à trouver la meilleure association entre deux groupes de valeurs : des **entrées** et des **cibles**. Par conséquent, le but de l'apprentissage supervisé est d'extrapoler les associations qui ont été trouvées à de nouvelles paires (entrée, cible) qui n'ont jamais été vues par le modèle. Puisque ce document s'intéresse à la prévision de séries chronologiques, cette section se limite à l'apprentissage supervisé dont la prévision fait partie.

L'apprentissage supervisé est utilisé quand nous voulons créer un modèle informatique qui « raisonne » de lui-même et apprend à réaliser une tâche bien précise. L'exemple typique est la classification d'images : à partir des pixels de l'image, déterminer dans quelle classe celle-ci se trouve. Par exemple, on pourrait chercher quel nombre est représenté dans l'image ou quel est l'objet que l'on peut observer. Ce type d'apprentissage automatique est appelé **classification** et tente de mettre en relation une entrée (une image, des informations tabulaires, un extrait audio, etc.) avec une **classe**. La classe est normalement représentée par un nombre discret ou encodée selon un vecteur indicateur.

Bien que la classification soit de loin ce qui est le plus utilisé dans le domaine, c'est la régression qui est le sujet principal de ce chapitre. La **régression** met en relation une entrée avec une cible dénotée par un nombre réel, voire un vecteur de nombres réels. Fait intéressant, il est relativement simple de transformer un problème de classification en un problème de régression. C'est d'ailleurs la méthode utilisée par les réseaux de neurones lorsqu'ils sont utilisés en tant que classificateur.

2.1 Définitions et notations

Dans cette section, on définit la régression suivant ce qui a déjà été présenté à propos des séries chronologiques. L'objectif est de trouver un modèle f qui associe une entrée $x \in \mathbb{R}^n$ à une cible $y \in \mathbb{R}^m$. La paire (x, y) dénote un **exemple** (nommée ainsi sous le thème de l'apprentissage par exemple) et collectivement, un ensemble d'exemples forment un **ensemble de données**.

Afin d'arriver à faire de la modélisation, il est admis que les exemples sont des échantillons d'une distribution de probabilité jointe $p(X, Y) \sim \mathcal{D}$ et que chaque exemple est indépendant des autres. Dans ce contexte, le but de l'apprentissage supervisé est de trouver une fonction $y = f(x)$ qui approxime la distribution conditionnelle $p(y|x)$.

Encore une fois, le modèle est choisi parmi une famille de modèles $\mathcal{F}$ et paramétré par un vecteur de **paramètres** noté θ. Si $\mathcal{L}$ est une fonction de cout qui permet d'évaluer l'erreur d'approximation, alors le modèle optimal est paramétré par

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(X,Y)}[\mathcal{L}(f(X;\theta), Y)].$$

Comme il est impossible (normalement) d'échantillonner directement de la distribution $\mathcal{D}$, il faut se contenter d'une approximation qui minimise l'erreur empirique calculée sur les N exemples de l'ensemble de données :

$$\hat{\theta}^* = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(f(x_i;\theta), y_i).$$

Malheureusement, il ne suffit pas de s'en tenir à cette définition.

2.2 L'optimalité est sous-optimale

Le processus d'entrainement tente de trouver une paramétrisation qui mène à une erreur de prédiction minimale. De cette façon, il est possible d'approximer de façon arbitrairement précise l'ensemble de données sur lequel le modèle est entrainé. Cette façon de faire est typique des problèmes de régression.

Cependant, à l'inverse des problèmes usuels de régression, le domaine de l'apprentissage automatique (et à fortiori celui de l'apprentissage profond) a tendance à surparamétrer ses modèles. Dans ces situations où le modèle est si puissant, il peut être relativement simple de trouver une paramétrisation qui puisse prédire parfaitement chaque exemple de l'ensemble de données. Malheureusement, ceci est un piège et mène à une mauvaise modélisation. Le problème vient du fait que ce modèle n'est entrainé que sur un échantillon limité de $\mathcal{D}$. Si un modèle arrive à parfaitement modéliser l'ensemble de données, il est plus que possible que celui-ci ait également appris le bruit inhérent à l'échantillonnage de l'ensemble de données. En d'autres mots, cela indique que le modèle ne pourra probablement pas bien généraliser à des données n'ayant jamais été vues par le modèle, même si elles sont échantillonnées dans $\mathcal{D}$. Ce problème est appelé le **surentrainement** et constitue un problème omniprésent dans les efforts de modélisation en intelligence artificielle.

Comme le problème vient de la difficulté de généraliser un modèle à des données n'ayant jamais été traitées pendant l'entrainement, une solution consiste à garder certaines données à l'abri du modèle pendant l'entrainement. Celles-ci ne sont utilisées que pour évaluer l'aptitude de généralisation du modèle. Cet ensemble de données mis de côté est appelé **ensemble de validation**, par opposition à l'**ensemble d'entrainement** utilisé pour optimiser le modèle. Cela est une première étape, mais qu'arrive-t-il si le modèle n'a pas de bonnes performances sur l'ensemble de validation ? Il faut retourner à la planche à dessin et construire un modèle plus approprié, souvent en changeant sa structure ou la façon dont il est entrainé. Comme cela revient souvent à changer des **hyperparamètres**, des paramètres externes aux modèles (c'est-à-dire qu'ils ne font pas partie de θ), pour avoir de bons résultats, cette méthode est sujette au même problème que l'optimisation automatique des paramètres sur l'ensemble d'entrainement : il est possible de souffrir de surentrainement même sur l'ensemble

de validation. La solution la plus reconnue reste la même : l'ensemble de validation est lui aussi séparé en deux, un ensemble (plus petit) de validation et un **ensemble de test**, ce dernier étant gardé que pour rapporter les valeurs finales des mesures de performances.

2.3 Les réseaux de neurones

Les réseaux de neurones [GBCB16] sont des modèles composés de **neurones** qui sont tout simplement une transformation affine d'un vecteur d'entrées composé avec une fonction non linéaire σ

$$y = \sigma\left(\sum_i w_i x_i + b_i\right).$$

Le coefficient w_i est appelé un **poids** du neurone et contrôle l'amplitude de la transformation. À la transformation linéaire est ajouté le **biais** b_i afin d'obtenir la transformation affine.

Une fonction non linéaire populaire est la courbe logistique, aussi appelée **sigmoïde**,

$$\sigma(z) = \frac{1}{1 + e^{-z}}.$$

La fonction sigmoïde peut également être utilisée afin de produire un nombre strictement entre zéro et un qu'il est possible d'interpréter en tant que probabilité. Poursuivant l'analogie, la fonction non linéaire représente le résultat de l'excitation d'un neurone par de l'énergie en entrée et par conséquent il est fréquent d'y référer en tant que **fonction d'activation**.

Un réseau de neurones est une collection de neurones mis ensemble. Dans sa version la plus simple, un réseau de neurones collecte les perceptrons et les évalue en parallèle

$$h_j = \sigma\left(\sum_i w_{i,j} x_i + b_i\right)$$
$$y = \sum_j v_j h_j + c_j$$

ou dans sa forme vectorielle maintenant couramment utilisée

$$y = V\sigma(Wx + b) + c.$$

L'étape intermédiaire $h = Wx + b$ est appelée la **couche cachée** ou latente du réseau de neurones. Les variables W, b, V et c du réseau forment, par concaténation, le vecteur de paramètres θ.

La plus simple des méthodes de mise à jour des poids, est la **descente de gradient stochastique**. Cette méthode met à jour les paramètres θ à l'itération k en leur ajoutant un vecteur de translation dans la direction opposée à celle de plus grande ascension (l'opposé du gradient)

$$\theta^{(k+1)} := \theta^{(k)} - \lambda \nabla_\theta \mathcal{L}(f(x;\theta), y).$$

Afin d'éviter à avoir à faire une recherche linéaire le long de ce vecteur de déplacement, un coefficient d'apprentissage λ est utilisé afin de réduire la taille du gradient et tenter d'éviter les problèmes d'oscillation. La partie stochastique découle du fait qu'il est inutile, voire pénalisant, de calculer le gradient sur l'entièreté de l'ensemble d'apprentissage et se contente donc d'un sous-ensemble appelé *mini batch*.

La flexibilité de cette approche vient du fait que l'algorithme d'optimisation est agnostique à la forme que prennent les sorties et la fonction de cout du réseau de neurones. Par conséquent, il est possible d'adapter ces deux éléments en fonction du problème. Face à un problème de régression, le plus simple est de mesurer la distance entre le vecteur d'éléments produit par le modèle $\hat{y}$ et le vecteur cible attendus y. Dans ce cas, la fonction de cout $\mathcal{L}$ se définit par

$$\mathcal{L}(\hat{y}, y) = \|\hat{y} - y\|^2.$$

Une fois la fonction minimisée, le modèle associe aux entrées x une valeur approximée $\hat{y}$ de la valeur y attendue. Si le modèle ne commet aucune erreur, l'optimisation est finie et parfaite (même si le modèle est probablement surentrainé).

En classification, le vecteur de valeurs de sortie est normalement traité comme étant les probabilités relatives des classes à prédire. Pour y arriver, la fonction d'acti-

vation **softmax** est appliquée à chaque sortie i du réseau afin de trouver la probabilité de la classe i parmi les K classes possibles.

$$\text{softmax}(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}.$$

Une fois que le modèle produit une distribution de probabilités, il est possible de l'entrainer en utilisant la technique d'estimation du maximum de vraisemblance. Celle-ci tente de trouver les paramètres θ^* qui maximisent la vraisemblance $p(\hat{y} = y|x;\theta) = f(x;\theta)$ des données y qui sont censées être un encodage 0-1 identifiant la bonne classe. Ceci est possible en minimisant la log-vraisemblance négative par rapport aux paramètres θ

$$\begin{aligned}
\theta^* &= \arg\max_\theta p(\hat{Y} = Y|X;\theta) \\
&= \arg\max_\theta \prod_i^N \prod_j^K p(\hat{y}_{i,j} = y_{i,j}|x_i;\theta) \\
&= \arg\max_\theta \sum_i^N \sum_j^K \log p(\hat{y}_{i,j} = y_{i,j}|x_i;\theta) \\
&= \arg\min_\theta -\sum_i^N \sum_j^K y_i \log f(x_i;\theta).
\end{aligned}$$

De cette façon, le modèle essaie d'associer une probabilité élevée aux exemples de cibles rencontrés dans l'ensemble de données. Avec assez de données, l'espoir est que le modèle ait vu assez d'exemples pour arriver à généraliser ses observations.

L'optimisation de la vraisemblance ne se limite pas qu'à la classification et fonctionne pour tous les **modèles génératifs**, c'est-à-dire les modèles qui produisent une distribution de probabilité conditionnelle caractérisant la distribution ayant généré les données. Soit g la distribution de probabilité paramétrée par $f(x;\theta)$, il suit que

$$\begin{aligned}
\theta^* &= \arg\max_\theta \sum_i^N \sum_j^K \log p(\hat{y}_{i,j} = y_{i,j}|x_i;\theta) \\
&= \arg\min_\theta -\sum_i^N \sum_j^K y_i \log g(y_i; f(x_i;\theta)).
\end{aligned}$$

Advenant que l'ensemble de test soit tiré de la même distribution que l'ensemble d'entrainement, il sera possible de prédire la distribution des valeurs possibles que y prendra. Cela implique que si une valeur complètement aberrante de x venait à apparaitre dans l'ensemble de test, il y a peu de chance que la prédiction soit la bonne. Par contre, si cette valeur avoisine les données vues dans l'ensemble de test, il est probable que le réseau puisse être généralisé à ces nouvelles données.

Il a été démontré [Cyb89] qu'un tel réseau de neurones peut approximer n'importe quelle fonction continue sur un sous-ensemble compact à $\mathbb{R}^n$ à une précision arbitrairement déterminée par le nombre de neurones présents. Ce résultat a motivé plus d'un chercheur à trouver comment utiliser ces réseaux de manière efficace.

Malheureusement, le nombre de neurones nécessaire dans la couche cachée pour atteindre une certaine précision croît très rapidement. L'alternative est d'enfiler les couches cachées de sorte que chaque couche cachée devienne l'entrée de la suivante

$$\begin{aligned} h_0 &= x \\ h_j &= \sigma(W_j h_{j-1} + b_j) \qquad j \in \{1, \ldots, J\} \\ y &= V h_J + c. \end{aligned}$$

Ces **réseaux de neurones profonds** permettent d'avoir une plus grande capacité qu'un réseau à une seule couche de taille vraiment plus grande [GBCB16]. Par contre, enchainer les transformations de cette manière mène à plusieurs problèmes qui ont trouvé solution au fil du temps.

2.3.1 Saturation des neurones

L'un de ces problèmes est la saturation des neurones. Un réseau de neurones est très utile pour modéliser des relations non linéaires entre les entrées et les sorties du réseau. De surcroit, ces relations peuvent être arbitrairement sensibles, c'est-à-dire qu'un faible changement à l'entrée peut provoquer une réaction en chaine qui peut changer radicalement la sortie. Par contre, cela a pour effet de parfois **saturer** un neurone lorsque celui-ci a des poids faisant en sorte que sa valeur de sortie est aux extrémités de la fonction d'activation.

Ce phénomène rend les mauvais comportements du réseau plus difficile à renverser. Plus l'entrée d'un neurone est de magnitude élevée, plus la valeur de sortie sera près de 0 ou 1. Par exemple, dans ce cas, le gradient de la fonction sigmoïde est pratiquement nul, même en cas d'erreur. Ceci rend un changement de comportement, voire un renversement, très difficile pour le modèle. De plus, si le modèle a été initialisé de façon à ce que ce neurone soit saturé dès le départ, cette tendance est d'autant plus difficile à renverser.

Afin de remédier à ce problème, il est possible d'utiliser des fonctions d'activation qui soient moins affectées par ce phénomène comme le *Rectified Linear Unit* (ReLU) [NH10] :

$$h(x) = \begin{cases} 0 & \text{si } x < 0 \\ x & \text{sinon} \end{cases}$$

et ses dérivées comme le Leaky ReLU [MHN13], PReLU [HZRS15], ELU [CUH15], ... qui sont maintenant utilisées de facto par les outils d'apprentissage profond.

2.3.2 Puissance de calcul

Le plus gros frein à l'adoption des réseaux de neurones dans les années 90 était dû à la puissance de calcul que ceux-ci requièrent. Or, la loi de Moore et la démocratisation des cartes graphiques sont partiellement responsables des avancées scientifiques en apprentissage profond. En traitement automatique de la langue, GPT-3 propose fièrement un modèle de 175 milliards de paramètres [BMR+20], un nombre impensable de paramètres il y a 14 ans quand l'apprentissage profond a fait surface.

En plus des avancées matérielles, il y a des développements qui ont contribué à l'utilisation efficace des réseaux de neurones. Avec le changement de fonction d'activation pour le ReLU le calcul de la non-linéarité s'effectue beaucoup plus rapidement que dans le cas de la sigmoïde (n'ayant ni division ni fonction exponentielle à calculer) en plus d'aider à empêcher la saturation des neurones.

2.3.3 L'instabilité du gradient

Depuis la redécouverte du mode inverse de la différentiation automatique[RHW86, Lin76, G$^+$89] sous le nom de la rétropropagation, les méthodes de descentes de gradient sont au coeur du processus d'apprentissage des réseaux de neurones. Malheureusement, la sensibilité qui crée la force d'expression du réseau de neurones est aussi un défi quand vient le temps de l'entrainer. Cet effet prend deux formes tout aussi dévastatrices [PMB13, BSF94] : l'explosion du gradient dans le cas où les changements dans les poids des couches inférieures réagissent démesurément aux valeurs du gradient ainsi que l'inverse où peu importe la valeur du gradient, le signal d'erreur se rendant aux couches inférieures est trop petit pour avoir un impact. En fait, le problème débute au moment de l'initialisation des poids du réseau [GB10]. Si celle-ci est faite sans soins, il est possible de se retrouver avec un modèle qu'il sera très difficile de ramener sur le chemin d'une optimisation efficace.

Ces problèmes peuvent être atténués de nombreuses façons. L'utilisation de *dropout* [HSK$^+$12], une méthode de normalisation des réseaux de neurones profonds où le signal de chaque neurone peut être ramené à zéro de façon aléatoire. Il y a également la *batch normalization* [IS15] une méthode de remise à l'échelle attaquant le problème du côté des entrées en les empêchant de trop varier sur leur domaine. Ou encore les nouvelles architectures de réseau comme le *ResNet* [HZRS16], une architecture basée sur les LSTMs (discuté dans la Section 2.4.1) et qui donne un court-circuit au signal du gradient [SGS15, JZS15] afin qu'il puisse se rendre à tous les niveaux du réseau.

2.3.4 Non-interprétabilité

Finalement, l'un des plus grands problèmes des réseaux de neurones (en dehors de leur complexité et de la science-alchimie [Hut18] qui les propulsent) est leur inhérente difficulté à être interprétés. Malgré leurs succès, le côté « boite noire » des réseaux de neurones profonds en effraie plus d'un quand il vient le temps de les mettre en production dans un système pouvant avoir des impacts réels sur la vie des gens.

Malgré les efforts investis à tenter d'expliquer les décisions d'un réseau de neurones [DBH18], le problème est loin d'être résolu. Il est d'ailleurs un sujet d'actualité scientifique centré sur la question de la confiance que l'on peut avoir en nos algorithmes

d'intelligence artificielle.

La non-interprétabilité des modèles, combinée avec la complexité liée à leur utilisation, a fait en sorte que les praticiens du domaine de la prévision de séries chronologiques ont boudé les réseaux de neurones jusqu'à tout récemment [GTS16].

2.4 Réseaux récurrents

Une variante intéressante des réseaux de neurones est le **réseau de neurones récurrent**. Ces réseaux sont en mesure de prendre en entrée des séquences de longueurs arbitraires ce qui les rend parfaits pour les séries chronologiques.

Les réseaux récurrents fonctionnent en utilisant la couche cachée du réseau de neurones de base comme un canal de communication à travers les pas de temps, tout en restant relativement légers puisqu'ils réutilisent la vaste majorité des poids pour chaque élément de la séquence. Voici leur définition plus formelle : soit $0 < t \leq T$ l'indice de la position dans la séquence, $0 < k < K$ le nombre de couches cachées et $h_{0,k}$ un hyperparamètre servant d'initialisation aux canaux de communication (souvent initialisé à zéro), alors

$$\begin{aligned} \hat{y}_t &= W_K h_{t,K-1} + b_K \\ h_{t,k} &= \sigma(W_k h_{t,k-1} + V_k h_{t-1,k} + b_k) \\ h_{t,0} &= x_t. \end{aligned}$$

Au moment de l'entrainement, ce réseau est déroulé en un très long graphe orienté acyclique qui permet d'utiliser la rétropropagation de l'erreur, mais en gardant en tête tous les problèmes liés à cette technique, surtout les problèmes de disparition ou d'explosion du gradient abordés à la Section 2.3.3.

2.4.1 Long Short-Term Memory

Vu le potentiel énorme de ce type de réseau pour la prévision de séries chronologiques, mais surtout pour le traitement automatique de la langue où les textes sont représentés par des séquences de mots, des efforts ont été investis à trouver une so-

lution au problème de propagation du gradient. La première qui ait vraiment gagné de la popularité dans la communauté (si ce n'est que bien après sa publication) est le modèle **LSTM** pour *Long Short-Term Memory Network* [HS97].

L'idée maitresse de ce modèle est l'utilisation d'un canal de communication en surplus de la couche cachée. Ce nouveau canal, appelé *cell* ou parfois la mémoire du LSTM, est affecté de portes logiques qui laissent passer ou non le signal. Dans les équations suivantes définissant un LSTM, $0 < t \leq T$ est l'indice de position dans la séquence et, malgré qu'il soit possible de définir un LSTM à plus d'une couche, l'indice de la couche est ici omis afin d'alléger la notation. Les portes d'entrée, de sortie et d'oubli, respectivement notées i_t, o_t, f_t sont définies par

$$\begin{aligned} i_t &= \sigma(W_i x_t + V_i h_{t-1} + U_i c_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + V_o h_{t-1} + U_o c_t + b_o) \\ f_t &= \sigma(W_f x_t + V_f h_{t-1} + U_f c_{t-1} + b_f) \end{aligned}$$

avec σ ici étant habituellement la fonction sigmoïde puisqu'elle permet de ramener la sortie de ces portes à des nombres entre 0 et 1. À noter que la porte de sortie se base sur le vecteur mémoire du pas de séquence courant plutôt que du précédent. Les portes d'oubli et d'entrée servent respectivement à contrôler la rétention de l'information à l'intérieur de la mémoire et de l'acquisition de nouvelle information :

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c x_t + V_c h_{t-1} + b_c).$$

La forme [1] de l'équation de la cellule rend ce second canal de communication plus efficace puisque la précédente cellule mémoire ne passe pas par la tangente hyperbolique et n'est affectée que d'une addition. Ceci simplifie énormément la rétropropagation du signal du gradient. En fait, en enlevant la porte d'oubli, on retrouve à quelques termes près le squelette d'un ResNet[HZRS16].

Finalement, le tout est combiné dans la production de la couche cachée

$$h_t = o_t \odot \tanh c_t$$

1. Le lecteur plus visuel est invité à consulter [Ola15] pour une explication pas-à-pas de chacune de ces équations aidé d'une représentation visuelle de celles-ci.

et la sortie est la même que pour un simple réseau récurrent

$$\hat{y}_t = W_y h_t + b_y.$$

Avec cette formulation qui mitige le problème de la disparition du gradient, il est possible de s'attaquer à des problèmes ayant des séquences de quelques centaines d'éléments. Ceci n'aurait jamais été efficace sans un meilleur flot du gradient ou sans une augmentation de la performance de calcul des ordinateurs.

2.4.2 Réseau Encodeur-Décodeur

Les réseaux du type encodeur-décodeur, communément appelés **Seq2Seq**, sont une extension simple, mais puissante des RNNs. En effet, construire un modèle Seq2Seq revient à construire deux réseaux de neurones récurrents, bout à bout : le premier pour encoder l'entrée du modèle et le second pour produire les sorties.

Cette idée a pris naissance dans le domaine du traitement automatique des langues où l'ordre des mots de la langue d'entrée était rarement l'ordre de la langue de sortie. Alors, afin de s'assurer d'avoir vu tous les mots avant de produire une traduction, le modèle a été séparé en deux parties [SVL14].

Cette façon de faire peut être adaptée au domaine de la prévision de séries chronologiques où il est possible de faire la distinction entre l'étape de l'encodage du contexte et la partie production de prévision. En séparant le modèle ainsi, il est possible d'utiliser deux structures un peu différentes, ce qui permet de gérer la disponibilité changeante des covariables telles qu'illustrées dans [WTNM17].

2.5 Flexibilité des réseaux profonds en prévision

Les modèles de prévisions statistiques sont tous construits minutieusement afin de permettre de représenter les données d'une façon bien précise. La classe des modèles ARMA [BDC02] se concentre sur l'utilisation de l'historique rapprochée et se base sur l'hypothèse que, malgré l'existence de chocs, le processus retourne toujours vers la moyenne plus ou moins rapidement (avec ou sans oscillations). La famille des modèles de lissage exponentiel ajoute de la complexité et de la capacité de représentation au

besoin afin de prendre en compte les effets de tendance et de saisonnalités, chacune des composantes fonctionnant un peu différemment.

Contrairement à ceux-ci, un point en faveur de l'apprentissage profond est sa flexibilité. Dans sa forme de base, le modèle est déjà un approximateur universel [Cyb89]. En pratique, cela veut dire que s'il y a assez de données, il est concevable d'accroitre le nombre de neurones, donc la capacité du modèle, afin de modéliser des relations complexes entre les entrées et les sorties. En outre, cela permet d'ajuster la capacité du modèle en fonction du problème et des données accessibles.

De plus, comme il est possible d'approximer n'importe quelle fonction continue, le modèle n'est plus limité aux fonctions linéaires, contrairement à la vaste majorité des modèles statistiques, ce qui ouvre la porte à la modélisation des relations non linéaires entre les différentes entrées du modèle.

Cette flexibilité a un prix. Plus la capacité d'un modèle est élevée, plus le besoin en données prend de l'importance afin d'entrainer correctement un modèle et en arriver à un niveau de confiance acceptable quant aux choix des paramètres [GBCB16]. Pour pallier ce problème, nous utilisons des architectures plus complexes de réseaux de neurones tels les réseaux profonds, les réseaux récurrents, les réseaux résiduels,... En encodant la forme du problème dans l'architecture du réseau, il est possible d'être plus efficace dans son utilisation des données [GBCB16].

2.5.1 Augmentation de la fréquence de mesure

La résolution des données quantifie la distance temporelle entre deux mesures de la série. Plus la résolution est basse, moins il y a de données disponibles et vice-versa. Les données à basse résolution peuvent être des mesures ponctuelles, par exemple la température à midi à chaque jour, ou agrégée comme la moyenne des températures ponctuelles horaires collectées au cours de la journée. Chacune de ces façons de présenter les données est valable, mais ne pourra pas être utilisée pour répondre aux mêmes questions (du moins, pas avec le même degré d'exactitude). Or, avec une résolution des données plus élevée (ici, horaire), l'analyste a la possibilité de choisir la représentation des données qui lui convient, voire d'utiliser les données ponctuelles horaires directement. Par récursivité, ce même argumentaire peut continuer d'être

appliqué jusqu'à l'obtention d'un flux continu de mesures offrant une flexibilité maximale (pour le cout modique d'un espace disque infini). Il convient donc, lors de la collecte des données, de faire un bon choix sur la granularité des mesures qui seront enregistrées.

Aussi tard qu'en 2018 lorsque la compétition M4 eut lieu [MSA20], les participants devaient produire des prévisions où le plus haut niveau de résolution était le niveau horaire. En parallèle, en robotique, il est courant de rencontrer des besoins en prévision ayant une résolution plus élevée qu'aux secondes afin qu'un robot soit en mesure d'évoluer dans un monde physique dynamique. Ce qui constitue une série à haute résolution est par conséquent dépendant du domaine d'application.

Il faut distinguer deux types de résolution : la résolution des mesures et la résolution des prévisions. Du point de vue de l'apprentissage profond, il est possible d'augmenter la résolution des mesures afin de produire de meilleures prévisions ayant une résolution comparativement plus basse [2]. Dans ce cas, la disponibilité de l'historique à un niveau de granularité plus fin permet une analyse supérieure de la distribution des données ainsi que la production d'un modèle qui reflète mieux la réalité. Ceci se traduit généralement par des prévisions ayant une variance réduite [GBCB16].

Cependant, cette façon d'augmenter les données a une limite : due à la lourdeur des modèles d'apprentissage profond, plus la fréquence des prévisions doit être élevée, moins il est possible, pour des raisons de temps de calcul, d'utiliser les modèles plus complexes pour des prévisions en temps réel.

2.5.2 Augmentation de la quantité de covariables

Un second axe d'augmentation des données est l'ajout de covariables. Ces données ne sont pas des mesures du phénomène à l'étude, mais représentent d'autres phénomènes qui ont le potentiel d'expliquer la valeur de la série, que ce soit par corrélation ou par causalité.

L'utilisation de covariables est importante afin d'exprimer correctement la distri-

2. Aucun avantage ne découle d'une augmentation de la résolution des mesures accompagnée également d'un besoin de résolution des prévisions plus élevée. Cette augmentation ne résulte qu'en un accroissement de l'échelle des données, éliminant tout avantage potentiellement obtenu par l'addition de nouvelles informations.

bution de probabilité ayant généré une série. En présence de deux séries similaires provenant respectivement de deux distributions de probabilité différentes, il est possible d'unifier les deux problèmes en considérant la distribution de probabilité conditionnelle à un facteur externe qui explique la différence entre les séries. Par exemple, le nombre de chandails bleus et jaunes vendus suit des courbes similaires, mais les chandails bleus sont plus populaires. Bien que la distribution marginale des chandails de couleurs différentes soit elle aussi différente, la distribution de probabilité du nombre de chandails vendus, conditionné par la couleur du chandail, est la même pour les deux articles.

En pratique, il est plutôt difficile de savoir si une covariable apporte de l'information utile au modèle, surtout si la relation n'est pas linéaire. Le plus simple (mais onéreux en temps) est d'encoder les connaissances d'un expert dans le modèle afin d'exploiter ces relations. Par contre, si cela s'avère difficile ou impossible et qu'il existe assez de données pour y arriver, un réseau de neurones avec assez de capacité peut déceler ces relations par lui-même. Encore mieux, si l'analyste a accès à un expert et à un important ensemble de données, il est possible de créer une architecture spécifique au problème en utilisant les réseaux de neurones. En d'autres termes, il existe un continuum de modèles qui dépendent de la disponibilité des données ou d'experts dans le domaine du problème allant du modèle complètement informé par un expert au modèle entièrement informé par les données.

En utilisant ces données externes, il est possible de reconnaitre les changements d'état d'un phénomène, ce qui se traduit généralement par un changement de distribution génératrice. Ce changement d'état peut représenter une période d'incertitude, la présence potentielle de valeurs atypiques, un changement graduel des saisons, etc.

Il est à noter que la clé reste le nombre d'exemples qu'un ensemble de données possède. Avec une seule série chronologique, même avec une multitude de covariables, le modèle risque de ne pas reconnaitre l'information ajoutée. Dans ces cas, filtrer les covariables peut s'avérer avantageux afin d'aider le modèle à s'entrainer.

2.5.3 Augmentation du nombre de séries en parallèles

Un problème fondamental des séries chronologiques est celui de l'unicité de la ligne du temps. Il est impossible d'obtenir plus d'un échantillon de la distribution qui produit les mesures du phénomène spécifiquement à l'étude puisqu'il est souvent impossible de reproduire les conditions qui paramétrisent le phénomène, sauf en cas d'expériences soigneusement contrôlées. Ceci limite naturellement le nombre de données pouvant être collectées.

Malgré tout, une deuxième façon d'augmenter le nombre de données disponibles pour analyse est de considérer plus d'une série chronologique à la fois. Ces séries doivent être indépendantes [3] et identiquement distribuées afin de pouvoir appliquer un même modèle à chacune de ces séries. Cette hypothèse est souvent plus facile à respecter s'il est possible d'expliquer les différences entre certaines séries à l'aide de covariables (et avec les complications que cela engendre, comme discuté dans la section précédente).

Ce cas de figure est particulièrement intéressant dans le domaine du commerce de détail où un même modèle peut répondre aux besoins de prévisions de plusieurs produits, chacun de ces produits servant de réalisation de la distribution qui a généré les séries. Avec ces multiples exemples, il devient plus facile de déceler avec confiance statistique les liens plus complexes entre les covariables ou les valeurs historiques et les valeurs futures de la série. Cependant, ces liens complexes demandent une capacité de représentation accrue des modèles utilisés. Ceci est plutôt difficile à contrôler avec les modèles statistiques (à moins de changer de modèle), alors qu'il est trivial de modifier l'architecture d'un réseau de neurones afin d'avoir accès à plus ou moins de capacité.

Bref, les réseaux de neurones peuvent vraiment contribuer au domaine des séries temporelles, mais seulement lorsque la complexité du problème le justifie. Dans le cas contraire, les modèles profonds tombent vite dans le piège du surentrainement [4]. Le

3. Si les séries ne sont pas indépendantes, on peut alors considérer les modéliser conjointement de façon multivariée.

4. Cet argument est désormais controversé. Au cours des dernières années, il a été découvert empiriquement qu'il semble exister un point d'inflexion à partir duquel un modèle à capacité plus élevé devient plus facile à généraliser une fois l'erreur d'entrainement éliminée[BHMM19]. D'autres travaux ont aussi souligné que cet effet est également présent en considérant le nombre de données

prochain chapitre explore une sous-tâche particulièrement adaptée à l'utilisation des réseaux de neurones profonds : la quantification de l'incertitude.

ou le temps d'entrainement d'un modèle[NKB+19]. Les chercheurs ne s'entendent toujours pas sur l'explication de ce phénomène.

Chapitre 3

Les prévisions probabilistes

Produire des prévisions signifie se projeter dans le futur et tenter d'utiliser le passé comme gabarit pour approximer ce à quoi le futur pourrait ressembler. Pour faire confiance à cette prévision, il faut présumer que le futur ne sera pas affecté de situations anormales (ou du moins, rien d'anormal qui n'ait jamais été observé dans le passé). Mathématiquement, cela revient à réitérer l'hypothèse que la série chronologique à l'étude est une réalisation d'un processus stationnaire (Section 1.1). C'est pour cela que le concept de stationnarité en série chronologique est le pendant de l'hypothèse d'indépendance et de distribution identique en apprentissage automatique : il permet d'extrapoler le modèle à des données qui n'ont pas encore été observées.

Cette hypothèse, à la base de la discipline, est souvent vraie lorsqu'il est question d'un phénomène physique où l'évolution du système suit des règles strictes. Dans ces cas, s'il est possible d'avoir toutes les informations pertinentes, il est possible d'extrapoler le comportement du phénomène. Par opposition, les humains ont cette fâcheuse tendance à s'adapter à leur environnement et changer leurs comportements, c'est-à-dire d'être non-stationnaires. Ils ont une mémoire qui leur permet de prendre des décisions qui reposent sur des informations pouvant aller très loin dans le passé. Malheureusement, cette mémoire n'est pas parfaite, ce qui rend leurs comportements encore plus difficiles à prédire. Ceci explique pourquoi la plupart des questions auxquelles les analystes sont amenés à répondre sont au sujet de phénomènes humains.

Ce chapitre se veut une exploration des méthodes de prévision probabiliste dans un contexte de modèles d'apprentissage profond. L'accent est sur la justesse des distri-

butions de probabilité qui sont produites afin de modéliser l'incertitude du processus générateur de la série temporelle.

3.1 L'incertitude

Le but premier de la prévision de séries chronologiques est de permettre la prise de décisions basée sur une analyse de l'historique d'un phénomène. Pour ce faire, un ou plusieurs modèles sont créés afin d'étendre au futur les observations du passé. Outre les cas triviaux, la complexité des phénomènes étudiés et des modèles utilisés fait en sorte que les résultats sont nécessairement des approximations. La différence entre les valeurs vraissemblables produites par le modèle et la « vérité » définissent l'erreur de modélisation. Celle-ci se répartit généralement en quatre composantes :

L'erreur de mesure Résultat du manque de précision dans la mesure rapportée du phénomène à l'étude. Par exemple, l'erreur de mesure d'un thermomètre utilisé pour suivre la température.

L'erreur d'échantillonnage Les données utilisées pour l'apprentissage du modèle ne sont souvent qu'un échantillon de l'ensemble des mesures possibles. Dans le contexte des séries chronologiques, cela se traduit surtout par le manque de résolution dans les mesures et l'absence de mesures pour les facteurs contribuant à l'évolution du phénomène. Pour en revenir à l'exemple de la température, la prise de mesures à intervalle horaire et l'absence de mesures de l'ensoleillement contribueraient à l'erreur d'échantillonnage. Le premier dû à une résolution assez faible des mesures et le second par manque d'informations contribuant à la valeur à prédire.

L'erreur structurelle Le choix du modèle joue un rôle critique pour la construction d'une bonne approximation. Un modèle trop simple n'a pas assez de marge de manœuvre pour approximer un phénomène complexe. À l'opposé, un modèle trop complexe aura tendance à essayer de capturer les artéfacts résultant de l'erreur de mesure et d'échantillonnage. La recherche d'hyperparamètres et de modèles ont pour objectif de minimiser cette erreur.

L'erreur d'estimation Le plus juste des modèles est bien peu utile s'il est trop dif-

ficile de l'ajuster aux données observées. L'erreur d'estimation est causée par la difficulté à optimiser un modèle afin d'approximer correctement les paramètres de la distribution de prédiction. Pour minimiser l'erreur d'estimation, les modèles sont optimisés pour minimiser la différence entre les valeurs produites par le modèle et les observations, aussi appelée la méthode d'estimation par maximum de vraissemblance.

Puisque, du point de vue de l'analyste, l'erreur de mesure et l'erreur d'échantillonage sont souvent hors de son contrôle, ce texte se concentre seulement sur l'erreur structurelle et d'estimation. Par conséquent, le but de la modélisation devient d'approcher le plus possible la distribution de laquelle les données sont échantillonnées. En d'autres termes, le but est de trouver le modèle pouvant décrire la distribution de probabilité qui maximise la vraisemblance de l'ensemble de données (d'entrainement) et de toutes autres données découlant du même phénomène (souvent représentées par un ensemble de données de test).

3.1.1 Quantifier l'incertitude

Peu importe la source de l'incertitude, le résultat est le même : les prédictions ponctuelles d'un modèle entrainé pour reproduire les mesures du phénomène ne sont pas exactes. Afin de quantifier cette inexactitude, un modèle produit une distribution de probabilité représentant l'éventail des valeurs que le modèle croit possibles ainsi que leur pondération. Cette distribution peut être utilisée pour quantifier l'erreur attendue de la prévision. Cette forme de prévision est appelée prévision probabiliste ou prévision en distribution. À noter qu'une prédiction ponctuelle, c'est-à-dire en un point plutôt qu'en distribution, est en fait une prévision en distribution déguisée où il n'est rapporté que l'espérance (ou parfois la médiane) de la distribution. Les distributions de choix pour les prévisions ponctuelles réelles sont les distributions normales (quand un modèle est entrainé avec la méthode des moindres carrés) et les distributions de Laplace (lorsque le modèle est optimisé en minimisant l'erreur absolue entre les prévisions et les mesures). Dans le domaine des séries à nombres entiers, les distributions de Poissons et binomiale négative reviennent souvent.

L'avantage des prévisions probabilistes est qu'il devient beaucoup plus facile de

prendre des décisions basées sur ces prévisions puisque le modèle fourni une mesure de l'incertitude de modélisation du phénomène. En d'autres termes, plus la variance de la distribution de prévision est grande, plus le modèle est incertain de la valeur réelle du phénomène. Ceci peut ensuite être utilisé par l'analyste afin de prendre des décisions plus éclairées.

Il existe plusieurs façon de générer ces prévisions probabilistes. La plus juste se base sur la modélisation Bayesienne [WH06, BCC11] et permet de définir pour chaque donnée en entrée et pour chaque paramètre du modèle une distribution représentant notre incertitude sur leurs valeurs. Une fois combiné, le modèle est en mesure de produire une distribution à postériori prenant en compte la majorité des sources d'incertitude. Malheureusement, cette façon de procéder est très onéreuse en calcul et oblige à faire des hypothèses plutôt contraignantes sur les distributions des différents éléments du modèle [Bis06]. Il est possible de se rabbatre sur des approximations, mais encore une fois le cout en calculs des méthodes MCMC [1] reste encore aujourd'hui prohibitif. Par conséquent, ces méthodes ne sont pas à l'étude dans ce document.

Une autre possibilité est d'approximer directement la distribution. Avec une famille de modèles assez puissante, le modèle est à même de considérer directement sa confiance envers les données en entrée et de produire des prévisions probabilistes en conséquence. En utilisant des réseaux de neurones profonds, il devient donc possible de produire des prévisions probabilistes basées sur des relations non linéaires. Puisque la sortie d'un réseau de neurones est un vecteur de valeurs, il faut réinterpréter ce vecteur en une distribution de probabilité. Pour ce faire, il existe plusieurs techniques qui sont explorées dans la prochaine section.

3.2 Modèles probabilistes de séries temporelles

Les techniques pour obtenir des prévisions probabilistes se divisent en deux catégories : les techniques paramétriques et les techniques non paramétriques. Dans le premier cas, l'incertitude est modélisée avec une distribution de probabilité spécifique. Les valeurs en sortie du modèle sont utilisées pour paramétrer la distribution et générer les statistiques associées. Le second cas ne demande pas à l'analyste de faire

1. Markov Chain Monte-Carlo

un choix sur la distribution et utilise les valeurs en sortie pour approximer certains points de la distribution. Avec assez de ces valeurs générées, il est possible de reconstruire une distribution de probabilité à une résolution plus ou moins élevée, selon les besoins.

Voyant la flexibilité de la version non paramétrique, il se doit de se demander pourquoi celle-ci n'est pas exclusivement utilisée. Le problème est le même que pour le choix d'un modèle : la technique non paramétrée est plus complexe et demande d'avoir accès à plus de données afin d'éviter le surentrainement. Par contre, si l'analyste a une connaissance suffisante du problème pour choisir judicieusement la distribution de probabilité, le nombre de degrés de liberté plus limité devient un atout pour les cas où le surentrainement est problématique. De plus, une distribution de probabilité donne une résolution infinie (une courbe continue) de la prévision probabiliste, ce qui peut être un atout selon la situation. Finalement, les distributions connues de probabilités ont souvent des propriétés intéressantes qui les rendent plus faciles à manipuler.

Or, dans le contexte à l'étude pour ce document, il est supposé que la quantité de données est adéquate pour entrainer un modèle complexe (tel un réseau de neurones profond) et par conséquent il est souhaitable que l'analyste ne se rabatte que sur le strict minimum de contraintes sur les données. Par conséquent, le seul modèle probabiliste qui est considéré est la mélange de Gaussienne puisque celle-ci permet d'approximer des distributions arbitrairement complexes.

Au niveau des techniques non paramétriques, les modèles de prévisions probabilistes de séries chronologiques utilisant des modèles neuronaux profonds ont connu un essor considérable dans les dernières années. Ce texte étudie les modèles et techniques les plus prometteurs se basant soit sur la régression quantile ou les techniques de Monte-Carlo.

3.2.1 Mélanges de gaussiennes

Se voulant une extension de la gaussienne, le mélange de gaussienne[2] est tout simplement une méthode qui permet d'exprimer des distributions qui ne sont pas

2. À noter que le choix de distribution pour un modèle de mélange ne se limite pas à la loi gaussienne. Cette dernière est seulement le choix le plus commun dû à ses propriétés intéressantes.

nécessairement unimodales en combinant un nombre fini de gaussiennes. Cette combinaison peut être arbitrairement complexe selon le nombre de gaussiennes combinées.

Le procédé est assez simple. Soit N le nombre de gaussiennes utilisées pour le mélange et $p_i > 0$ le poids associé à chacune de ces gaussiennes de sorte que $\sum_{i=0}^{N} p_i = 1$. La distribution est donc

$$GM(y \mid [\mu_i, \sigma_i, p_i]_{i=1}^{N}) = \sum_{i=1}^{N} p_i \mathcal{N}(y|\mu_i, \sigma_i).$$

Pour échantillonner de cette distribution, il suffit d'abord d'échantillonner d'une distribution multinomiale paramétrée par $\{p_i\}_{i=0}^{N}$. Le résultat de cet échantillon identifie de quelle gaussienne il faut subséquemment échantillonner, cette dernière étant paramétrée par le couple (μ_i, σ_i) approprié.

Pour utiliser la méthode du mélange de gaussienne, un réseau de neurones n'a qu'à produire en sortie les paramètres $\{[p_i, \mu_i, \sigma_i]_{i=0}^{N}\}$. L'entrainement se fait en utilisant la fonction de log vraisemblance négative calculée sur la somme pondérée de toutes les distributions gaussiennes. Un désavantage de la méthode reste son instabilité numérique par rapport à l'optimisation par descente de gradient. Une possible solution à ce problème est de tronquer la magnitude du gradient afin de se protéger contre les divisions par des valeurs avoisinant zéro [GBCB16].

L'avantage de cette méthode est qu'elle ne dépend que de relativement peu de paramètres. De par son ubiquité, la distribution gaussienne est souvent la distribution de choix lorsque l'analyste n'a que peu d'information sur le problème et le mélange de gaussienne en est une extension naturelle. L'entrainement se fait relativement facilement et à faible cout (aucun besoin de méthodes de Monte-Carlo) et l'échantillonnage se fait également simplement. Le plus gros désavantage de la méthode est qu'il est plutôt difficile de trouver un quantile précis de la distribution (par exemple pour faire un intervalle de confiance à 95%) sans passer par une méthode plus onéreuse d'approximation.

Pour trouver le quantile α d'un mélange de distribution f_i, on suppose que leur fonction de répartition F_i est connue et facilement calculable. Par conséquent, la

fonction de répartition du mélange est

$$F_M(x) = \sum_i p_i F_i(x)$$

et le quantile α est la valeur

$$F_M^{-1}(\alpha) = \inf_x F_M(x) > \alpha.$$

Cette valeur peut se trouver avec un peu de travail en utilisant une recherche linéaire de type diviser pour régner, mais elle est également souvent approximée par échantillonnage.

3.2.2 Régression quantile

En revenant à la question initiale, il est intéressant de réaliser que, dans l'optique où un analyste souhaite utiliser des prévisions probabilistes afin d'estimer le risque de ses prévisions, il n'est pas vraiment nécessaire d'avoir accès à l'ensemble de la distribution de probabilité : souvent, seules quelques valeurs de repères, les quantiles, sont nécessaires.

Le **quantile** α de la fonction de répartition F est la valeur qui minimise

$$F^{-1}(\alpha) = \inf_x \{x \in \mathbb{R} | F(x) \geq \alpha\}.$$

En d'autres termes, c'est la valeur séparant le domaine de la fonction de masse de probabilité de sorte que $\alpha\%$ de la masse se trouve avant ce point.

Identifier les quantiles sans se rabbatre sur une forme analytique de distribution de probabilité ne se fait pas directement. Pour ce faire, une méthode ne supposant pas une forme fonctionnelle fixe pour la distribution a été proposée par [KBJ78]. Initialement, cette méthode se voulait être une méthode non paramétrique utilisée pour évaluer des quantiles spécifiques d'une distribution de probabilité en sortie d'un modèle linéaire. Plutôt que de faire une hypothèse de loi de probabilité, l'astuce est d'approximer la valeur des quantiles directement en changeant la fonction de cout à optimiser. Par conséquent, il est possible de modéliser les quantiles en ayant une

sortie du modèle par quantile d'intérêt. Les quantiles les plus intéressants sont :

— (2.5, 97.5) et (5, 95) pour leur représentation respective des intervalles de confiance à 95% et 90%,

— (25, 75) qui modélise l'interval dans lequel la moitié des valeurs se trouvent,

— (50), la médiane.

La difficulté majeure pour trouver ces valeurs provient du fait que les valeurs cibles pour chacun des quantiles ne sont pas disponibles pour comparaison directe (contrairement à la moyenne ou la médiane où l'échantillon sert de valeur cible). De surcroit, il ne faut surtout pas confondre les quantiles de la distribution marginale $p(y)$ avec les quantiles de la distribution conditionnelle $p(y|x)$. Bien que ces premiers soient faciles à calculer (il n'y a qu'à calculer les statistiques sur l'ensemble des valeurs de la série, sans se préoccuper des valeurs en entrée), ce sont ces derniers qui apportent de l'information sur laquelle il est possible de prendre des décisions.

Approcher les quantiles sans avoir de valeurs cibles est possible en biaisant judicieusement la mesure de l'erreur absolue, ce qui l'incline d'un côté ou de l'autre, selon s'il faut approcher un quantile inférieur ou supérieur à la médiane ($\alpha = 0.5$). Sa forme quelque peu particulière (Figure 3.1 lui a valu l'appellation en anglais de la *pinball loss* qui est traduite dans ce document par l'erreur absolue inclinée [3], définie par l'équation suivante :

$$L_\alpha(y, \hat{y}) = \alpha \max(0, y - \hat{y}) + (1 - \alpha) \max(0, \hat{y} - y).$$

De cette formulation, il est possible de retrouver la mesure d'erreur absolue moyenne à partir de l'égalité $\max(0, a) = \frac{1}{2}(a + |a|)$:

$$\begin{aligned} L_\alpha(y, \hat{y}) &= \alpha \max(0, y - \hat{y}) + (1 - \alpha) \max(0, \hat{y} - y) \\ &= \frac{\alpha}{2}(y - \hat{y} + |y - \hat{y}|) + \frac{(1 - \alpha)}{2}(\hat{y} - y + |y - \hat{y}|) \\ &= \alpha(y - \hat{y}) + \frac{1}{2}(\hat{y} - y + |y - \hat{y}|) \\ &= (\alpha - \frac{1}{2})(y - \hat{y}) + \frac{1}{2}|y - \hat{y}|. \end{aligned}$$

3. Cet auteur refuse d'appeler cette fonction « erreur machine à boule. »

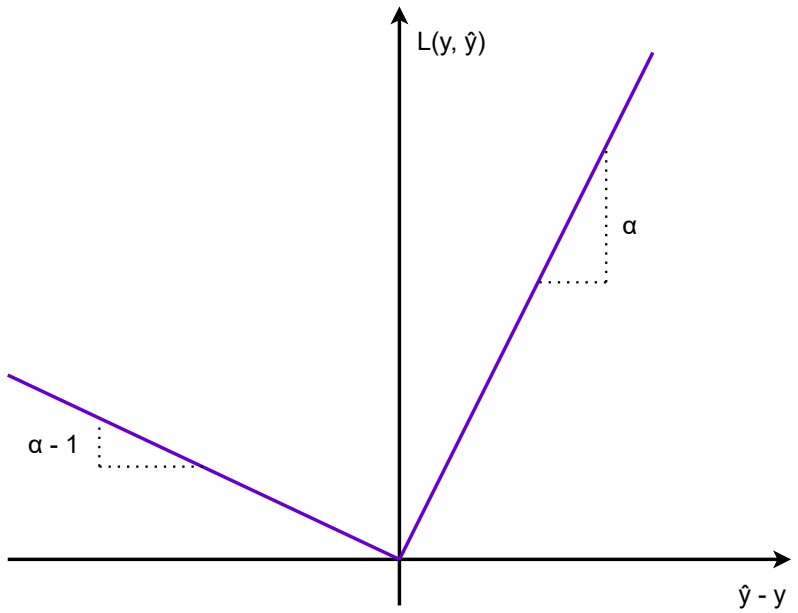

Figure 3.1 – Illustration de la fonction d'erreur absolue inclinée. La pente de chacun des côtés dépend du quantile recherché.

Donc, pour la médianne en $\alpha = \frac{1}{2}$, $\arg\min L_{\frac{1}{2}}(y, \hat{y}) = \arg\min \frac{1}{2}|y - \hat{y}| = \arg\min |y - \hat{y}|$.

Pour être pleinement efficace, l'analyste tentera d'approximer les quantiles d'intérêt pour le phénomène à modéliser et les décisions qu'il souhaite rendre possibles d'évaluer. Par exemple, un concept important en gestion de portefeuille est la valeur à risque (VaR $p\%$), c'est-à-dire qu'un portefeuille affiche un risque de perdre au moins cette valeur dans seulement $p\%$ des cas. Un exemple analogue est celui de l'inventaire de sureté, c'est-à-dire l'inventaire qui doit être gardé afin de s'assurer de ne plus avoir d'articles que dans $q\%$ des cas. Ces deux exemples représentent des situations où une analyste souhaiterait estimer, respectivement, le quantile $\alpha = p$ et $\alpha = 1 - q$. Chacun de ces quantiles se situe généralement près des extrémités de l'intervalle $(0, 1)$, endroit où les distributions comme la gaussienne ont tendance à moins bien représenter les données (un phénomène appelé distributions à queues épaisses). Dans ces cas, l'estimation directe des quantiles contourne le problème en approximant directement (et sans faire d'hypothèse sur la distribution) la valeur du quantile voulu. La fonction de cout à optimiser devient donc la somme des erreurs absolues inclinées pour chaque quantile agrégée par la somme ou la moyenne sur l'ensemble des pas de temps de

l'horizon de prévision.

Cette méthode ne suppose aucune distribution de probabilité et est très efficace au niveau du cout de calcul. Sa plus grande lacune est l'impossibilité d'utiliser directement les valeurs des quantiles pour échantillonner de la distribution que ceux-ci définissent. Bien que cela ne soit pas nécessaire pour calculer un niveau de confiance des prévisions du modèle, cela empêche tout de même d'utiliser la sortie du modèle directement pour l'échantillonnage de trajectoire décrit à la prochaine section, ce qui limite la modélisation aux modèles directs de prévision. Une autre lacune concerne les quantiles aux extrêmes de la distribution : plus un quantile est loin de la médiane, moins, par définition, y a-t-il de données de l'autre côté de ce quantile pouvant servir de contrepoids au reste de l'ensemble de données. Par conséquent, ces quantiles sont plus difficiles à estimer correctement et ont tendance à être sous-estimés (c'est-à-dire être trop près de la médiane). Outre l'acquisition de plus de données, la seule solution au problème est d'introduire des hypothèses contraignants l'espace de recherche des quantiles.

Pour retrouver la capacité d'échantillonner la distribution, il suffit d'interpoler les valeurs entre les quantiles pour reconstruire la fonction de répartition inverse $F^{-1}(\alpha)$. Afin de se simplifier la tâche, il est possible d'augmenter le nombre de quantiles prédits, réduisant ainsi l'erreur d'approximation induite par le choix de la méthode d'interpolation.

En augmentant le nombre de quantiles prédits, les valeurs des quantiles ont naturellement tendance à se rapprocher les unes des autres. Cela peut entrainer un problème de croisement des quantiles, c'est-à-dire qu'un quantile au niveau α a une valeur plus élevée qu'un autre quantile au niveau $\alpha + \varepsilon, (0 < \varepsilon < 1 - \alpha)$, ce qui est manifestement impossible de par la définition des quantiles. Heureusement, les modèles décrits dans ce document peuvent contourner ce problème grâce à leur flexibilité de modélisation. L'astuce est de définir chacun des quantiles en fonction de leurs distances avec les autres. Par exemple, il est possible de faire une prédiction de la médiane ($\alpha = 0.5$) et de s'en servir comme point d'ancrage sur lequel toutes les autres valeurs sont déduites. Soit f un modèle d'apprentissage paramétré par θ, x l'entrée, $\hat{y}$ la prédiction et $\{\alpha_i > 0.5\}$ l'ensemble des quantiles supérieurs à la médiane de telle

sorte que $\alpha_0 = 0.5$ et $\alpha_{i+1} > \alpha_i$. Il s'en suit que

$$\begin{aligned}\hat{y}_{\alpha_0} &= f(x, \theta, \alpha_0 = 0.5) \\ \hat{y}_{\alpha_i} &= \hat{y}_{\alpha_{i-1}} + \varepsilon_i \\ &= \hat{y}_{\alpha_{i-1}} + f(x, \theta, \alpha_i) \\ &= \sum_{j=0}^{i} f(x, \theta, \alpha_j).\end{aligned}$$

Afin d'éviter les problèmes de croisement, il suffit de s'assurer que les valeurs $f(x, \theta, \alpha_i)$ pour $i \neq 0$ soient positives, par exemple en utilisant la fonction *softplus*

$$\text{softplus}(x) = \ln(1 + e^x)$$

qui assure que la sortie est strictement positive.

Une stratégie similaire, mais plus sophistiquée, est employée par [GBW+19]. Ils utilisent des splines linéaires isotoniques, c'est-à-dire une fonction continue définie par morceaux où chaque morceau est une fonction linéaire monotone croissante. Sous sa forme générale, la spline linéaire isotonique se définie par

$$s(\alpha; \gamma, \mathbf{b}, \mathbf{d}) = \gamma + \sum_{l=0}^{L} b_l \max(0, \alpha - d_l).$$

En somme, γ est la probabilité pour une valeur de la distribution sous-jacente aux quantiles d'être inférieur à d_0. Les sommes cumulatives des b_l sont les pentes de chacun des morceaux. Les valeurs d_l servent à indiquer où chacun des morceaux débute. Encore une fois, il faut s'assurer que les valeurs d_l soient ordonnées, ce qu'il est possible de réaliser en reparamétrant la valeur $d_l = \sum_{j=1}^{l} \delta_j$, pour $0 \leq \delta_j \leq 1$ et $\sum_j \delta_j = 1$. Voir la Figure 3.2 pour un résumé de la notation.

Pour s'assurer que la spline soit monotone croissante, il faut s'assurer que la pente du morceau débutant par d_l soit positive, c'est-à-dire que $\sum_{j=1}^{l} b_l \geq 0$. En reparamétrant $b_l = \beta_l - \beta_{l-1}$ pour $\beta_0 = 0$ et $\beta_l \geq 0$, les pentes deviennent les β_l directement (la pente est $\sum_j b_j = \sum_j \beta_j - \beta_{j-1} = \beta_0 + \beta_l = \beta_l$) et de par leur restriction seront

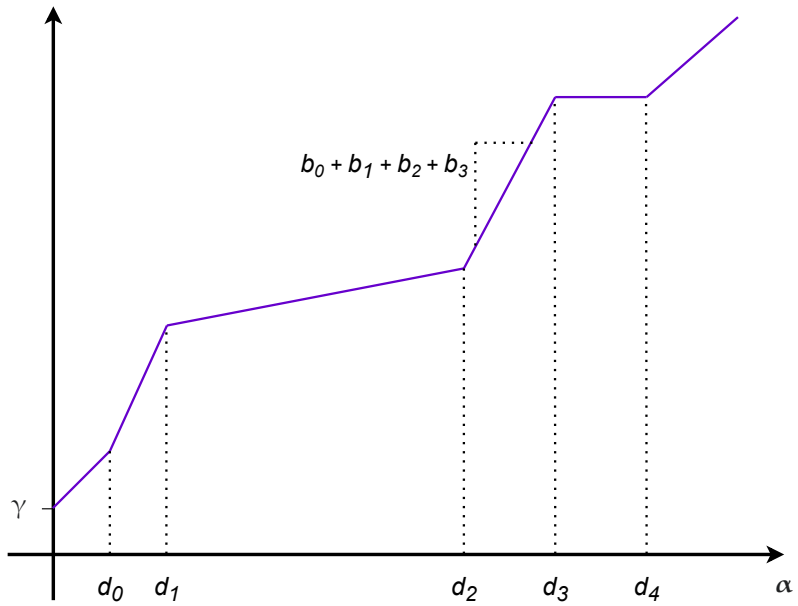

Figure 3.2 – Représentation visuelle de la spline isotonique. La fonction a pour valeur initiale γ et change de pente à chaque fois que α croise une nouvelle valeur dans l'ensemble des d_l. La pente du segment l est donnée par la somme $\sum_{i=0}^{l} b_i$.

toujours positives. Au final, la fonction

$$s(\alpha; \gamma, \beta, \delta) = \gamma + \sum_{l=0}^{L} (\beta_l - \beta_{l-1}) \max(0, \alpha - \sum_{j=0}^{l} \delta_j)$$

permet d'échantillonner la fonction de densité sous-jacente (sans jamais n'avoir à l'exprimer directement) seulement en mettant des restrictions sur $0 \leq \beta_j$ et $0 \leq \delta_j$, ce qui est simple à réaliser avec la fonction *softplus* mentionnée précédemment et une renormalisation servant à ramener la somme des δ_j à 1.

Entrainer un modèle utilisant cette représentation des quantiles est plus complexe que dans le cas où seulement des points discrets de la fonction quantile étaient disponibles [4]. Comme la fonction $s(\alpha; \cdot)$ permet de générer les valeurs des quantiles pour chacun des niveaux α, il faut calculer la fonction d'erreur absolue inclinée sur

4. En fait, il serait possible d'échantillonner des niveaux de quantiles aléatoires et de n'utiliser que ceux-ci pour calculer la fonction de perte. Malheureusement, cela laisse sur la table le plus gros avantage de cette méthode, c'est-à-dire une fonction continue décrivant les quantiles.

l'ensemble de son domaine :

$$L(y, s(\alpha; \cdot)) = \int_0^1 2L_\alpha(y; s(\alpha; \cdot))d\alpha.$$

Heureusement, comme la fonction s est une fonction linéaire isotonique par morceau, il existe une forme analytique à cette intégrale. Soit $\bar{\alpha}$ le niveau de quantile satisfaisant $s(\bar{\alpha}; \gamma, \mathbf{b}, \mathbf{d}) = y$ et est donné par

$$\bar{\alpha} = \frac{y - \gamma + \sum_{l=0}^{l_0} b_l d_l}{\sum_{l=0}^{l_0} b_l} l_0 = \max\{l \mid s(d_l; \gamma, \mathbf{b}, \mathbf{d}) < y, 0 \leq l \leq L\}.$$

alors, il s'en suit que

$$\begin{aligned}
&\int_0^1 2L_\alpha(y, s(\alpha; \gamma, \mathbf{b}, \mathbf{d}))d\alpha \\
&\quad = \int_0^1 2(\alpha \max(0, y - s(\alpha; \gamma, \mathbf{b}, \mathbf{d})) + (1 - \alpha)\max(0, s(\alpha; \gamma, \mathbf{b}, \mathbf{d}) - y))d\alpha \\
&\quad = \int_0^{\bar{\alpha}} 2\alpha(y - s(\alpha; \gamma, \mathbf{b}, \mathbf{d})) + \int_{\bar{\alpha}}^1 2(1 - \alpha)(s(\alpha; \gamma, \mathbf{b}, \mathbf{d}) - y)d\alpha \\
&\quad = (2\bar{\alpha} - 1)y + (1 - 2\bar{\alpha})\gamma + \sum_{l=0}^{L} b_l \left(\frac{1 - d_l^3}{3} - d_l - \max(\bar{\alpha}, d_l) + 2\max(\bar{\alpha}, d_l)d_l \right)
\end{aligned}$$

L'utilisation de la fonction d'erreur absolue inclinée et de son extension continue la fonction quantiles par splines linéaires isotoniques permet d'entrainer n'importe quel modèle à faire non seulement des prévisions, mais également de produire les quantiles de la distribution de probabilité représentant l'incertitude du modèle envers ses propres prédictions. Les deux permettent d'échantillonner facilement de nouvelles valeurs en générant des échantillons uniformes sur $[0, 1]$ et en les projetant dans le domaine de la distribution apprise à l'aide de la fonction quantile $F^{-1}(\alpha)$. Le revers de la médaille est qu'il est difficile de produire les statistiques normalement disponibles en utilisant un modèle paramétrique comme l'espérance et la variance de la distribution, c'est pourquoi la plupart des modèles utilisant les méthodes par quantiles se contentent généralement de rapporter la médiane en tant que prévision ponctuelle plutôt que l'espérance conditionnelle.

3.2.3 L'échantillonnage de trajectoires

Une fois le choix de la méthode d'approche des quantiles trouvée, soit l'approche paramétrique ou l'approche non paramétrique, l'étape suivante est de trouver le modèle qui saura produire les paramètres de la technique le plus efficacement possible. Dans le cas des séries chronologiques, il faut donc se questionner sur le type d'approche à privilégier entre les prévisions directes et les prévisions récursives (Section 1.2.1).

Dans le cas des prévisions directes, il suffit de multiplier le nombre de valeurs de sortie requis par la technique d'approche des quantiles par le nombre de pas de temps nécessaire pour couvrir l'horizon de prévision. Dans le cas de la prévision récursive, il suffit de prime abord de prédire la valeur de la série (et les quantiles mesurant l'incertitude du modèle) pour le prochain pas de temps.

Malgré que les prévisions récursives semblent être la solution la plus logique, elle a été critiquée pour sa tendance à créer des prévisions qui ont des intervalles de confiance trop petits [WTNM17]. À moins d'être en présence d'un phénomène hautement régulier, il est usuel de s'attendre à constater des intervalles de confiance grandissants à mesure que l'on s'éloigne des valeurs connues vers le futur. Or, en utilisant la technique commune de *teacher forcing* [WZ89] où la valeur correcte de la série est réutilisée en entrée pour le modèle, ce dernier apprend qu'il aura toujours des valeurs à jour en entrée, se libérant ainsi de composer son incertitude d'un pas de temps à l'autre. En d'autres termes, il faut distinguer les prédictions successives avec des valeurs mises à jour $p(y_{t+h}|y_{0:t+h-1}; x_{0:t+h-1})$ des prédictions récursives $p(y_{t+h}|y_{0:t}, \hat{y}_{t:t+h-1}; x_{0:t+h-1})$ où les prévisions du modèle entre les pas de temps t et $t+h$ sont utilisées [5].

DeepAR [SFGJ20] utilise cette stratégie avec succès pour la prévision de la demande de produits chez Amazon. L'idée clé est d'utiliser un LSTM afin de produire, à chaque pas de temps, les paramètres d'une distribution de probabilité. L'article suggère d'utiliser une loi binomiale négative pour modéliser la demande en valeurs discrètes positives et la loi t-Student pour la modélisation des valeurs réelles. Cette distribution est apprise en utilisant la technique de *teacher forcing* pendant l'entrainement. Par contre, en inférence, les valeurs échantillonnées à chaque pas de temps

5. Il est possible d'utiliser le même traitement avec les covariables.

sont retournées en entrée du modèle pour le pas de temps suivant. La clé est que la distribution de probabilité utilisée pour mesurer la confiance du modèle en ses prédictions n'est pas donnée par les distributions paramétrées à chaque pas de temps, mais plutôt en tirant un certain nombre (10 000 dans leur article) de trajectoires aléatoires et en calculant les quantiles obtenus par cet échantillon. Bien entendu, produire 10 000 prévisions plutôt qu'une seule requiert plus de puissance de calcul, mais comme cela n'est nécessaire qu'au moment de l'inférence, ce cout est relativement insignifiant avec les ordinateurs d'aujourd'hui, surtout si c'est traité en parallèle [6].

La méthode d'échantillonnage de trajectoire peut être utilisée autant avec les modèles paramétriques qu'avec les modèles non paramétriques, pourvu qu'il soit possible de générer des échantillons dans un temps raisonnable. Elle a d'ailleurs été reprise par [GBW+19] qui la combine avec la fonction quantiles de splines isotoniques et [WTNM17] qui remplace le LSTM par un modèle Seq2Seq augmenté d'un réseau de neurones profond.

3.3 Méthodologie

Une analyste désire modéliser un phénomène temporel afin de prendre une décision d'affaires. Afin de compléter son travail, elle doit faire une analyse des enjeux liés aux risques associés à la possibilité que la série chronologique à l'étude prenne des valeurs inusitées dans le futur. Si l'intervalle des valeurs attendues (l'intervalle de confiance) est surestimé, la décision risque d'être trop conservatrice. S'il est sous-estimé, la décision pourrait mener à une situation où les conséquences auront plus d'impact que prévu. Les méthodes ponctuelles de prévisions ne sont pas appropriées et celles d'estimation d'intervalle de confiance n'ont pas porté fruit, car il semble que la distribution de probabilité du processus stochastique soit inhabituelle. Il est possible d'avoir assez de données pour entrainer un réseau de neurones. Dans une telle situation, il peut être plus simple d'attaquer le problème directement avec une famille de modèles plus expressifs et flexibles tels que les réseaux de neurones profonds. À noter que cette décision dépend de plusieurs facteurs et que la disponibilité de données

6. À noter que certaines distributions sont beaucoup plus gourmandes quand vient le temps de générer des échantillons.

et la complexité du problème en sont que deux parmi tant d'autres. Ce document n'a pas pour objectif de démontrer la suprématie des réseaux de neurones profonds autant qu'il trahit le type de modèles avec lesquels l'auteur est le plus confortable.

Dans cette optique, ce chapitre évalue les techniques d'estimation de la distribution des prévisions sur deux axes principaux : leur précision et leur flexibilité. Ce dernier est un axe d'analyse entièrement qualitative et sert seulement de contrepoint à l'analyse de la précision. Un axe manquant est celui du cout de calcul. Sachant que ces techniques sont utilisées en conjonction avec un réseau de neurones profond, il est supposé que l'entrainement et l'évaluation du réseau dominent le nombre de calculs et sont donc exclus des critères d'évaluation.

Le but est d'être en mesure de faire des recommandations sur le type de techniques à privilégier selon le type de modélisation à effectuer à la fin de cette section.

3.3.1 Données

Pour évaluer les techniques, trois ensembles de données ont été choisis pour leur disponibilité en ligne, leur apparition dans des publications antérieures et pour leur volume de données.

Le premier est appelé *Traffic* [DG17]. Cet ensemble de données représente 15 mois de données pris du site web du Département des Transports de la Californie. Les données décrivent avec un nombre entre 0 et 1 le taux d'occupation de plusieurs voies automobiles de la région de la baie de San Francisco. Le second, appelé *Electricity* [DG17] est une collection de la consommation en électricité de 321 clients au Portugal. Les données sont agrégées à l'heure. Finalement, le dernier ensemble de données provient de la compétition M4 [MSA20] duquel ne sont gardées que les séries à valeurs horaires afin d'avoir une résolution des données appropriée pour l'application de l'apprentissage profond. Ces séries regroupent des données de sources diverses, dont l'économie, la finance et l'internet des objets. Les trois ensembles sont présentés dans le Tableau 3.1 avec quelques caractéristiques.

Les données sont normalisées dans la bande $[-1, 1]$ par transformation linéaire qui associe la valeur minimum de la série à -1 et la valeur maximum à 1. Le résultat est ensuite passé aux modèles et dénormalisé avant de calculer les mesures de perfor-

Nom	Nombre de séries	Nombre moyen de valeurs	Résolution	Horizon
Traffic	862	14 036	Horaire	24
Electricity	321	21 044	Horaire	24
M4 Hourly	414	854	Horaire	24

Tableau 3.1 – Caractéristiques des trois ensembles de données à l'étude. Chacun présente des séries chronologiques avec des mesures horaires, ce qui permet d'avoir une résolution plus granulaire et un nombre assez élevé de données pour entrainer un réseau de neurones.

mances. Afin de simplifier l'entrainement des modèles récurrents et non récurrents, les données sont découpées en fenêtres du quadruple du nombre de pas de temps dans l'horizon de prévision, ici 96 dans les trois cas. Les fenêtres sont ensuite échantillonnées au hasard pendant l'entrainement. L'ensemble de test est lui composé d'un unique horizon de prévision pour chaque série temporelle.

3.3.2 Modèles et techniques

Le Chapitre 2 a passé en revue les modèles les plus couramment utilisés pour faire de la prévision de séries chronologiques. Chacun attaque le problème de façon un peu différente selon que le modèle effectue des prévisions directes ou s'il est souhaité de faire des prévisions récursives. Ce chapitre ne vise pas à couvrir extensivement l'ensemble des combinaisons possibles de modèles et de techniques de prévisions probabilistes. Il se concentre plutôt sur les approches les plus prometteuses telles qu'informées par les récentes publications où elles sont apparues. Par conséquent, les modèles spécifiques à la gestion des covariables ou des valeurs manquantes sont ignorés. Les modèles ici adressent plutôt la façon de modéliser les données en entrées plutôt que la forme des sorties nécessaire à l'utilisation des techniques de régression aux quantiles.

Afin de bien évaluer les techniques d'approche des quantiles présentées dans la Section 3.2, chacune est testée avec différents types de réseaux de neurones. Les modèles choisis, décrits au Chapitre 2 sont :

— Le réseau de neurones profond simple (*feed-forward* ou FFNN),
— Le LSTM : DeepAR [SFGJ20] et
— Seq2Seq : MQ-RNN [WTNM17].

Ces choix permettent d'avoir une vue d'ensemble des possibilités utilisant les réseaux de neurones, sans entrer dans l'exploration des modèles niches à un certain domaine ou tentant de résoudre un certain problème. Les modèles et leur choix d'hyperparamètres sont présentés au Tableau 3.2.

Modèle	couches	neurones	activation	p dropout	déplacement
FFNN	2	40	ReLU	0	10^{-4}
LSTM	2	40	tanh	0.1	10^{-3}
Seq2Sec	1	50	tanh	0.1	10^{-3}

Tableau 3.2 – Les hyperparamètres choisis pour chacun des modèles. Ceux-ci ont été ajustés dans le but d'avoir de bons résultats sur l'ensemble de données de M4 à l'aide d'une recherche d'hyperparamètres aléatoire et reproduit pour les deux autres. La colonne « déplacement » décrit le pas de déplacement, communément appelé le *learning rate*. Les modèles sont également tous utilisés en conjonction avec *dropout* [HSK+12], une technique de réduction du bruit de la surface d'optimisation des paramètres.

À partir de ces modèles, on évalue chaque technique, présentée avec ses propres hyperparamètres dans le Tableau 3.3, avec chacun des modèles auxquels ils se rattachent.

Technique	Hyperparamètres	Modèles
mélange de gaussiennes	3 gaussiennes	FFNN, LSTM
Régression Quantile	[0.05, 0.1, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.75, 0.8, 0.9, 0.95]	Sec2Sec
Splines isotoniques	8 morceaux	FFNN, LSTM

Tableau 3.3 – Techniques d'approximation des quantiles, leurs hyperparamètres principaux ainsi que les modèles testés avec cette technique.

En plus de ces résultats, deux autres modèles récemment publiés sont également inclus dans les résultats. Le premier est *Deep Factors* [WSM+19] utilisé conjointement avec les méthodes de mélanges et de splines. Le modèle est basé sur un LSTM ayant une couche cachée de 40 neurones et 10 facteurs. Le modèle local se limite à 5 neurones pour ne pas surcharger le nombre d'hyperparamètres. Le second modèle est *Deep State Space* [RSG+18] utilisé avec la méthode directe de régression quantile. Ce modèle utilise également le LSTM comme modèle de base, deux couches cachées de

40 neurones et un taux de *dropout* de 10%. Ces modèles ne sont pas l'objet principal de l'étude, mais ont tout de même été ajoutés dans un désir d'exhaustivité.

Tous ces modèles ont été implémentés avec GluonTS [ABBS+20], une bibliothèque de modèles de séries temporelles publiées par Amazon et regroupant plusieurs implémentations de modèles récents de prévisions basées sur l'apprentissage profond. Chaque modèle est entrainé sur 300 époques en utilisant Adam [KB14] et une initialisation des paramètres *Xavier* [GB10].

3.3.3 Évaluation

Pour évaluer la performance de prévision des modèles entrainés, l'erreur absolue moyenne calibrée a été utilisée puisque tous les modèles sont en mesure de donner une prévision basée sur la médiane.

Afin de mesurer la précision des techniques d'approche de la fonction quantiles, il faut faire preuve d'ingéniosité. En effet, il est impossible de connaitre les « vraies » valeurs des quantiles conditionnées sur la valeur des entrées. Les chercheurs du domaine ont donc trouvé des façons alternatives de mesurer la précision des quantiles.

La plus simple est d'utiliser directement la mesure d'erreur absolue inclinée puisqu'il a été démontré que celle-ci est minimale que si les quantiles sont correctement prédits. Ceci permet en effet de comparer les valeurs entre les différents modèles, mais pas de savoir si, dans l'absolu, une méthode a une bonne ou une mauvaise performance. Afin d'éviter l'effet des séries à plus grands nombres que d'autres, l'erreur absolue inclinée calibrée est rapportée. Cette calibration n'est qu'une remise à l'échelle par division de la cible réelle [7].

Une autre mesure est la couverture quantile qui évalue si les quantiles prédits couvrent effectivement les cibles dans une proportion égale à leur niveau de quantile. Pour mesurer la couverture, il suffit de calculer la proportion des exemples où la cible tombe sous la valeur du quantile prédit

$$\text{couverture}_\alpha(\mathbf{y}, \hat{\mathbf{y}}^{(\alpha)}) = \frac{1}{|\mathbf{y}|} \sum_{i}^{|\mathbf{y}|} \mathbf{1}(y_i < \hat{y}_i^{(\alpha)})$$

7. Advenant le cas où toutes les valeurs de la série seraient à 0, la version non calibrée est retournée.

où **1** est la fonction indicatrice. Si $\alpha\%$ des exemples ont leur valeur cible inférieure à la valeur du α-quantile prédit, la couverture est parfaite. Pour évaluer une fonction quantile complète, il faut choisir quelques points de la fonction et agréger les résultats. Dans ce document, la moyenne des erreurs absolues de la couverture calculée pour chaque décile

$$\text{couverture}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{9} \sum_{i=1}^{9} \left| \text{couverture}_{\frac{i}{10}}(\mathbf{y}, \hat{\mathbf{y}}^{(\frac{i}{10})}) - \frac{i}{10} \right|$$

est rapportée. À noter que la couverture est calculée sur les quantiles des données cibles sans égards aux valeurs d'entrée. En d'autres termes, les quantiles marginaux sont ceux évalués, pas les quantiles conditionnels qui sont les valeurs d'intérêt.

3.4 Résultats

Cette section rapporte les résultats des expériences décrites à la section précédente. Tous les modèles sont entrainés à trois reprises et le meilleur modèle (en validation) est choisi pour l'analyse. À travers toutes les combinaisons, le réseau de neurones profond (FFNN) est le modèle ayant affiché le plus d'instabilité quant à ses résultats et son entrainement.

D'abord, on retrouve au Tableau 3.4 l'erreur absolue moyenne calibrée des modèles sur les trois jeux de données à l'étude. Pour chacun des modèles, la médiane de la distribution de sortie est utilisée en tant que prédiction du modèle. Pour les modèles utilisant un mélange de gaussienne en sortie, la médiane est approximée à l'aide d'un échantillon de 100 valeurs en provenance de la distribution. Pour les autres, le quantile 0.5 est utilisé.

Ensuite, au Tableau 3.5 se trouve la moyenne des erreurs absolue de couverture (Section 3.3.3) des déciles (les quantiles 0.1, 0.2, ..., 0.9). Encore une fois, les quantiles du mélange de gaussienne sont approximés via un échantillon de 100 valeurs de sortie du modèle. Il faut analyser ces résultats avec prudence puisque la couverture, bien qu'un bon guide pour s'assurer que les résultats ne sont pas complètement faux, compare les quantiles conditionnés par les valeurs en entrée (ici l'historique des données chronologiques) avec les quantiles marginaux du jeu de donnée. Si les quantiles

	Traffic		Electricity		M4 Hourly	
	Mélange	Splines	mélange	Splines	Mélange	Splines
FFNN	1.0862	1.1071	1.0184	0.9846	1.5456	1.6089
LSTM	0.5993	**0.5872**	0.7420	**0.6590**	**1.2137**	1.2840
Deep Factor*	11.544	11.548	1.2534	1.2515	12.451	12.388
Seq2Seq	1.5798		2.0082		4.7458	
Deep State Space*	2.0220		1.2660		1.5947	

Tableau 3.4 – Erreur absolue moyenne calibrée des prévisions pour les ensembles de données décrits dans la Section 3.3.1. Les modèles annotés d'un * sont rapportés à titre indicatif seulement.

marginaux sont parfaitement prédits, alors la moyenne d'erreur de couverture absolue est de zéro. Par contre, la réciproque n'est pas nécessairement vraie.

	Traffic		Electricity		M4 Hourly	
	Mélange	Splines	Mélange	Splines	Mélange	Splines
FFNN	0.0333	**0.0192**	0.0296	0.0223	**0.0065**	0.0197
LSTM	0.0456	0.0456	0.0568	0.0241	0.0614	0.0848
Deep Factor*	0.3497	0.3586	0.1359	0.1361	0.0492	0.0501
Seq2Seq	0.1092		**0.0219**		0.0866	
Deep State Space*	0.0878		0.0938		0.1657	

Tableau 3.5 – Erreur absolue moyenne de la couverture des déciles pour les ensembles de données décrits dans la Section 3.3.1. Les modèles annotés d'un * sont rapportés à titre indicatif seulement.

Finalement, afin d'avoir une meilleure vue d'ensemble de l'évaluation des quantiles générés par le modèle, le tableau 3.6 rassemble les mesures de la moyenne calibrée des erreurs absolues inclinées. Cette fois encore, la mesure doit être interprétée avec prudence : bien qu'il soit démontré que les quantiles prédits sont exacts si et seulement si les erreurs absolues inclinées sont minimales [KBJ78], il est impossible de savoir quelle est la valeur minimale ou même de supposer que les conditions d'optimalité sur l'ensemble d'entrainement se généralisent à l'ensemble d'évaluation.

Une analyse visuelle d'exemples aléatoires révèle que le problème avec *Deep Factor* est l'incapacité du modèle de reconnaitre la saisonnalité dans les données. D'ailleurs, bien que l'article [WSM+19] relate des résultats sur *Electricity* qui indiquent qu'il soit possible d'utiliser le modèle global pour capturer les effets de saisonnalité, il a

	Traffic		Electricity		M4 Hourly	
	Mélange	Splines	Mélange	Splines	Mélange	Splines
FFNN	0.2206	0.2354	0.0757	0.0734	0.0428	0.0383
LSTM	0.1268	**0.1236**	0.0531	**0.0454**	0.0354	**0.0262**
Deep Factor*	1.6847	1.6799	0.4064	0.4085	0.1237	0.1247
Seq2Seq	0.3359		0.1283		0.0522	
Deep State Space*	0.3873		0.1381		0.0341	

Tableau 3.6 – Erreur absolue moyenne calibrée des erreurs absolues inclinées pour les ensembles de données décrits dans la Section 3.3.1. Les modèles annotés d'un * sont rapportés à titre indicatif seulement.

été impossible de reproduire ceux-ci avec l'implémentation disponible.

3.5 Analyse

Le but de ce document est de trouver vers quelles méthodes un analyste devrait aller en premier lieu lorsque vient le temps d'utiliser les méthodes d'apprentissage profond pour un problème de prévision de séries chronologiques. Les résultats de la section précédente offrent quelques réponses à cet égard.

Le Tableau 3.4 illustre de meilleurs résultats pour le modèle récurrent à travers chacun des jeux de données dans son approximation de la médiane, utilisée comme prévision (plutôt que la moyenne) par tous les modèles. Cela coïncide avec les résultats de l'erreur absolue inclinée pour le quantile $\alpha = 0.5$ (non rapportés), peu importe la méthode d'approximation des quantiles utilisée. Le même effet est observable au Tableau 3.6 où, encore une fois, le modèle récurrent semble le plus approprié vis-à-vis de la moyenne calibrée des erreurs absolues inclinées. Finalement, les résultats sur la couverture placent les modèles récurrents au second rang, ce qui confirme les performances de cette méthode.

La question subséquente concerne l'utilisation des mélanges de gaussiennes ou des splines. Dans presque tous les cas rapportés à la section précédente, les splines semblent avoir un léger avantage, mais la différence n'est pas aussi importante que celle attribuable au modèle.

En seconde position se trouve le réseau de neurones profond simple. Ce modèle a

l'avantage de ne pas être récurrent, rendant son entrainement plus simple et rapide comparativement aux modèles récurrents. Son plus gros désavantage est d'avoir à prédéfinir la longueur de l'historique qui se retrouve en entrée. Il se trouve que cela n'est pas un énorme désavantage quand il faut faire de même pour les méthodes récurrentes afin de réduire l'empreinte mémoire du modèle, mais il reste que le modèle récurrent est plus flexible. Un élément intéressant du simple réseau de neurones profond est qu'il arrive à une meilleure couverture que le modèle récurrent. Ceci pourrait être expliqué par le fait que de par son architecture, le réseau de neurones non récurrent arrive plus facilement à capturer la distribution globale des quantiles, contrairement au réseau récurrent qui est lui limité à une visibilité locale de la série en entrée. Cependant, il semble que cette même capacité nuise à la performance du réseau dans sa prédiction des quantiles conditionnés. Il faudrait par contre une analyse plus poussée afin de trancher sur la question.

Au sujet de la couverture, les résultats rapportés au Tableau 3.5 favorisent nettement le réseau de neurones profond simple. Or, ceci est également le cas pour *Deep Factor* qui a déjà été discrédité pour son incapacité à modéliser les effets de saisons. Ce dernier, surtout sur *M4-Hourly*, arrive à avoir des résultats semblables aux autres modèles malgré ses résultats médiocres selon les autres mesures. Ceci n'est qu'une autre preuve que la mesure de la couverture doit être utilisée qu'avec grands soins.

Finalement, la méthode d'encodeur-décodeur *Seq2Seq* ne semble pas arriver à apprendre suffisamment pour produire des résultats utiles. Malgré la recherche d'hyperparamètre, il semble que le modèle ne soit pas approprié à ce type de séries.

Au final, dans la pratique, ces résultats suggèrent qu'il est possible de s'en tenir aux méthodes de base tels que le réseau de neurones profond simple ou le LSTM. Vu le plus gros investissement en temps d'entrainement requis pour les modèles récurrents, il faut décider si le jeu en vaut la chandelle avant de choisir le LSTM à tout prix. Par son architecture, l'encodeur-décodeur pourrait être plus approprié en présence de séries chronologiques ayant des covariables.

En ce qui concerne le choix entre le mélange de gaussienne et les splines, la décision revient à l'utilisation que l'analyste veut faire du modèle : la méthode des splines ne donne pas d'approximation de la moyenne de la distribution des prévisions, ce qui peut être un problème dans certains cas. Par contre, le mélange requiert un échantillonnage

répété de la distribution afin d'approximer les quantiles, mais peut avoir accès à la moyenne directement. Dans les deux cas, l'approximation des quantiles semble dépendre plutôt du modèle utilisé pour générer les paramètres que de la méthode de représentation des quantiles.

Malheureusement, il est impossible de tirer de vastes généralisations à partir de cette étude. Ces expériences ne se limitent qu'aux séries chronologiques multiples univariées de longueurs importantes. Elles ne touchent pas les méthodes à utiliser en présence de covariables, si l'horizon de prédiction devient plus long (par exemple, quelques mois pour une série à résolution horaire), s'il y a des valeurs manquantes ou si le modèle doit être réentrainé régulièrement. Toutefois, il semble que les méthodes les plus simples restent les plus appropriées dans ce cas spécifique, ce qui devrait rassurer l'analyste qui désire débuter son étude à l'aide de ces modèles, même sur les jeux de données plus complexes.

Conclusion

Malgré l'histoire houleuse entre les communautés d'apprentissage automatique et de prévision de séries chronologiques, ce document se veut une exploration de ce que les réseaux de neurones sont en mesure d'apporter au domaine de la prévision de séries temporelles. L'apprentissage profond, malgré ses succès récents, ne devrait pas être vu comme une panacée pouvant remplacer toutes les autres méthodes, mais plutôt comme un nouvel outil de l'arsenal de l'analyste qui dérive sa puissance non pas de la structure induite sur les données ou le modèle, mais de l'abondance des données pouvant être collectées aujourd'hui à l'ère d'internet.

Pourquoi est-ce que le lissage exponentiel, voire la méthode de Holt-Winters est maintenant l'un des premiers outils que les prévisionnistes utilisent ? D'une certaine façon, c'est que les analystes ont appris, au fil des années, que ce sont les méthodes les plus simples et efficaces qui donnaient de bons résultats [GTS16]. En d'autres termes, ils ont appris que, pour un certain type de données et peu importe la distribution sous-jacente de la série, un modèle bien optimisé de Holt-Winters donne des résultats satisfaisants pouvant être raffinés par la suite, si nécessaire.

Le Chapitre 3 se veut une évaluation semblable pour le cas des séries univariée sans covariables pour les prévisions probabilistes. Il se trouve que les modèles plus simples d'apprentissage profond sont parfaitement en mesure de capturer l'information pertinente en ce qui a trait à la saisonnalité et à la forme des séries. [8] Joint à la méthode des splines linéaires isotonique, cette paire constitue un outil polyvalent et puissant pouvant servir de méthode de prévision des quantiles sans avoir à gérer un modèle qui soit trop instable.

8. Par expérience, même si ceci n'est pas l'objectif ici, cela se complique en présence d'effets de tendance.

3.6 Travaux futurs

Une première question concerne l'utilisation des prévisions probabilistes. De prime abord, l'utilisation de ce type de prévisions est d'une très grande valeur pour la planification robuste de processus, c'est-à-dire la planification avec un souci d'éviter (ou du moins de caractériser) le risque associé aux décisions qui sont prises. Encore une fois, à quoi ressemblerait l'algorithme d'optimisation qui devrait être joint à ceux mentionnés dans ce document pour arriver à un tel résultat ?

Sinon, il est possible de pousser plus loin l'étude des méthodes de prévisions. Avec les vastes possibilités qui s'ouvrent une fois que les réseaux de neurones sont acceptés comme un modèle valable pour la tâche de prévision, la présente étude semble bien simple. Juste sur le sujet des prévisions probabilistes, il peut y avoir des extensions à d'autres types de problèmes comme l'addition de covariables, les données à résolutions multiples ou les problèmes de prévisions sans historiques, pour ne nommer que ceux-là.

Il est fascinant (et un peu effrayant) de réaliser à quel point il existe une multitude de types différents de problème de prévision, selon le format des données en entrée. Un chercheur ne peut que se sentir humble face à la diversité des problèmes potentiels en lien avec la tâche pourtant simple d'extrapoler une série chronologique dans le temps.

En plus d'explorer d'autres types de problèmes, il pourrait être intéressant de tenter de trouver la frontière passée laquelle il devient plus intéressant de considérer les modèles plus complexes tels les réseaux de neurones plutôt que les modèles plus simples comme Holt-Winters ou ARIMA.

Qu'en est-il des problèmes d'application des modèles de prévisions ? Comment faut-il gérer le réentrainement des modèles, les points de changement, l'entrainement des modèles en présence de téraoctets de données, etc.

Ou peut-être qu'il serait plus approprié de trouver une association entre les spécificités du problème analysé et l'architecture idéale d'un réseau de neurones. De cette façon, le réseau pourrait être façonné par les caractéristiques des données afin de maximiser les chances de surmonter le problème à l'étude. Pour ce faire, il faudrait créer une librairie de problèmes divers ayant des complexités et des complications différentes et variées.

3.6. Travaux futurs

En fait, il est possible de développer une taxonomie des problèmes liés à la prévision qui puisse servir de guide sur le type de modèle ou de transformation des données qui soit nécessaire pour une famille de problème spécifique. Bien identifier un problème sera toujours le premier pas afin de le surmonter.

Bibliographie

[ABBS+20] A. Alexandrov, K. Benidis, M. Bohlke-Schneider, V. Flunkert, J. Gasthaus, T. Januschowski, D. C. Maddix, S. Rangapuram, D. Salinas, J. Schulz *et al.*, « GluonTS : Probabilistic and Neural Time Series Modeling in Python, » *Journal of Machine Learning Research*, vol. 21, no. 116, pp. 1–6, 2020.

[ACH18] S. Arora, N. Cohen, et E. Hazan, « On the Optimization of Deep Networks : Implicit Acceleration by Overparameterization, » 2018.

[AHM51] K. J. Arrow, T. Harris, et J. Marschak, « Optimal inventory policy, » *Econometrica : Journal of the Econometric Society*, pp. 250–272, 1951.

[BCC11] D. Barber, A. T. Cemgil, et S. Chiappa, *Bayesian time series models*. Cambridge University Press, 2011.

[BDC02] P. J. Brockwell, R. A. Davis, et M. V. Calder, *Introduction to time series and forecasting*. Springer, 2002, vol. 2.

[BHMM19] M. Belkin, D. Hsu, S. Ma, et S. Mandal, « Reconciling modern machine-learning practice and the classical bias–variance trade-off, » *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, pp. 15 849–15 854, 2019.

[Bis06] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[BJ70] G. E. P. Box et G. M. Jenkins, *Time series analysis ; forecasting and control*. San Francisco : Holden-Day, 1970.

[BMR+20] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, « Language models are few-shot learners, » *arXiv preprint arXiv :2005.14165*, 2020.

[Bro04] R. G. Brown, *Smoothing, forecasting and prediction of discrete time series*. Courier Corporation, 2004.

[BSF94] Y. Bengio, P. Simard, et P. Frasconi, « Learning long-term dependencies with gradient descent is difficult, » *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[Cro72] J. D. Croston, « Forecasting and stock control for intermittent demands, » *Journal of the Operational Research Society*, vol. 23, no. 3, pp. 289–303, 1972.

[CUH15] D.-A. Clevert, T. Unterthiner, et S. Hochreiter, « Fast and accurate deep network learning by exponential linear units (elus), » *arXiv preprint arXiv :1511.07289*, 2015.

[Cyb89] G. Cybenko, « Approximation by superpositions of a sigmoidal function, » *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[DBH18] F. K. Došilović, M. Brčić, et N. Hlupić, « Explainable artificial intelligence : A survey, » dans *2018 41st International convention on information and communication technology, electronics and microelectronics (MIPRO)*. IEEE, 2018, pp. 0210–0215.

[DG17] D. Dua et C. Graff, « UCI Machine Learning Repository, » 2017. Disponible à http://archive.ics.uci.edu/ml

[G+89] A. Griewank *et al.*, « On automatic differentiation, » *Mathematical Programming : recent developments and applications*, vol. 6, no. 6, pp. 83–107, 1989.

[GB10] X. Glorot et Y. Bengio, « Understanding the difficulty of training deep feedforward neural networks, » dans *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[GBCB16] I. Goodfellow, Y. Bengio, A. Courville, et Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[GBW+19] J. Gasthaus, K. Benidis, Y. Wang, S. S. Rangapuram, D. Salinas, V. Flunkert, et T. Januschowski, « Probabilistic forecasting with spline quantile

function rnns, » dans *The 22nd International Conference on Artificial Intelligence and Statistics*, 2019, pp. 1901–1910.

[GTS16] M. Gilliland, L. Tashman, et U. Sglavo, *Business forecasting : Practical problems and solutions*. John Wiley & Sons, 2016.

[HK06] R. J. Hyndman et A. B. Koehler, « Another look at measures of forecast accuracy, » *International journal of forecasting*, vol. 22, no. 4, pp. 679–688, 2006.

[HS97] S. Hochreiter et J. Schmidhuber, « Long short-term memory, » *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[HSK+12] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, et R. R. Salakhutdinov, « Improving neural networks by preventing co-adaptation of feature detectors, » *arXiv preprint arXiv :1207.0580*, 2012.

[Hut18] M. Hutson, « Has artificial intelligence become alchemy ? » 2018.

[HZRS15] K. He, X. Zhang, S. Ren, et J. Sun, « Delving deep into rectifiers : Surpassing human-level performance on imagenet classification, » dans *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[HZRS16] K. He, X. Zhang, S. Ren, et J. Sun, « Deep residual learning for image recognition, » dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[IS15] S. Ioffe et C. Szegedy, « Batch normalization : Accelerating deep network training by reducing internal covariate shift, » *arXiv preprint arXiv :1502.03167*, 2015.

[JZS15] R. Jozefowicz, W. Zaremba, et I. Sutskever, « An empirical exploration of recurrent network architectures, » dans *International conference on machine learning*, 2015, pp. 2342–2350.

[KB14] D. P. Kingma et J. Ba, « Adam : A method for stochastic optimization, » *arXiv preprint arXiv :1412.6980*, 2014.

[KBJ78] R. Koenker et G. Bassett Jr, « Regression quantiles, » *Econometrica : journal of the Econometric Society*, pp. 33–50, 1978.

[Lin76] S. Linnainmaa, « Taylor expansion of the accumulated rounding error, » *BIT Numerical Mathematics*, vol. 16, no. 2, pp. 146–160, 1976.

[MH00] S. Makridakis et M. Hibon, « The M3-Competition : results, conclusions and implications, » *International journal of forecasting*, vol. 16, no. 4, pp. 451–476, 2000.

[MHN13] A. L. Maas, A. Y. Hannun, et A. Y. Ng, « Rectifier nonlinearities improve neural network acoustic models, » dans *ICML*, vol. 30, 2013, p. 3.

[MMH20] P. Montero-Manso et R. J. Hyndman, « Principles and Algorithms for Forecasting Groups of Time Series : Locality and Globality, » 2020.

[MSA18] S. Makridakis, E. Spiliotis, et V. Assimakopoulos, « Statistical and Machine Learning forecasting methods : Concerns and ways forward, » *PloS one*, vol. 13, no. 3, p. e0194889, 2018.

[MSA20] S. Makridakis, E. Spiliotis, et V. Assimakopoulos, « The M4 Competition : 100,000 time series and 61 forecasting methods, » *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020.

[NH10] V. Nair et G. E. Hinton, « Rectified linear units improve restricted boltzmann machines, » dans *ICML*, 2010.

[NKB+19] P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, et I. Sutskever, « Deep double descent : Where bigger models and more data hurt, » *arXiv preprint arXiv :1912.02292*, 2019.

[OCCB20] B. N. Oreshkin, D. Carpov, N. Chapados, et Y. Bengio, « N-BEATS : Neural basis expansion analysis for interpretable time series forecasting, » dans *International Conference on Learning Representation*, 2020.

[Ola15] C. Olah, « Understanding LSTM Networks, » https://colah.github.io/posts/2015-08-Understanding-LSTMs/, 2015, accédé le 2020-12-01.

[PMB13] R. Pascanu, T. Mikolov, et Y. Bengio, « On the difficulty of training recurrent neural networks, » dans *International conference on machine learning*, 2013, pp. 1310–1318.

[RHW86] D. E. Rumelhart, G. E. Hinton, et R. J. Williams, « Learning representations by back-propagating errors, » *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[RSG+18] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, et T. Januschowski, « Deep state space models for time series forecasting, » dans *Advances in neural information processing systems*, 2018, pp. 7785–7794.

[SFGJ20] D. Salinas, V. Flunkert, J. Gasthaus, et T. Januschowski, « DeepAR : Probabilistic forecasting with autoregressive recurrent networks, » *International Journal of Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[SGS15] R. K. Srivastava, K. Greff, et J. Schmidhuber, « Training very deep networks, » *Advances in neural information processing systems*, vol. 28, pp. 2377–2385, 2015.

[SVL14] I. Sutskever, O. Vinyals, et Q. V. Le, « Sequence to sequence learning with neural networks, » dans *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[TH14] S. B. Taieb et R. Hyndman, « Boosting multi-step autoregressive forecasts, » dans *International Conference on Machine Learning*. PMLR, 2014, pp. 109–117.

[WH06] M. West et J. Harrison, *Bayesian forecasting and dynamic models*. Springer Science & Business Media, 2006.

[WSM+19] Y. Wang, A. Smola, D. C. Maddix, J. Gasthaus, D. Foster, et T. Januschowski, « Deep factors for forecasting, » *arXiv preprint arXiv :1905.12417*, 2019.

[WTNM17] R. Wen, K. Torkkola, B. Narayanaswamy, et D. Madeka, « A multi-horizon quantile recurrent forecaster, » *arXiv preprint arXiv :1711.11053*, 2017.

[WZ89] R. J. Williams et D. Zipser, « A learning algorithm for continually running fully recurrent neural networks, » *Neural computation*, vol. 1, no. 2, pp. 270–280, 1989.